A PARETO FRONTIER INTERSECTION-BASED APPROACH FOR EFFICIENT MULTIOBJECTIVE OPTIMIZATION OF COMPETING CONCEPT ALTERNATIVES

A Thesis
Presented to
The Academic Faculty

by

Damon A. Rousis

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
School of Aerospace Engineering

Georgia Institute of Technology
August 2011

A PARETO FRONTIER INTERSECTION-BASED APPROACH FOR EFFICIENT MULTIOBJECTIVE OPTIMIZATION OF COMPETING CONCEPT ALTERNATIVES

Approved by:

Dr. D. N. Mavris, Advisor
School of Aerospace Engineering
Georgia Institute of Technology

Dr. J. I. Jagoda
School of Aerospace Engineering
Georgia Institute of Technology

Dr. Vitali Volovoi
School of Aerospace Engineering
Georgia Institute of Technology

Dr. Brian German
School of Aerospace Engineering
Georgia Institute of Technology

Mr. Jeff Berton
Multidisciplinary Design, Analysis and Optimization Branch
NASA Glenn Research Center

Date Approved: June 30, 2011

ACKNOWLEDGEMENTS

I would first like to thank my advisor, Dr. Dimitri Mavris for giving me the opportunity to learn and grow at ASDL and always reminding me how high the bar really is. Thanks also to Dr. Jeff Jagoda, Dr. Brian German, Dr. Vitali Volovoi, and Jeff Berton for offering insight and spending the time as members of my thesis committee.

Thanks to Justin Gray at NASA Glenn Research Center for contributing many ideas and destroying my thesis topic multiple times daily. Along those same lines, I would like to thank the many people at ASDL who I befriended and would not hesitate to offer words of advice and/or criticism whether solicited or not.

Thank you, Marie for going through this journey with me and for your never ending support. And finally, thanks go to my parents who above all taught me to never settle for mediocrity.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

SUMMARY

The expected growth of civil aviation over the next twenty years places significant emphasis on revolutionary technology development aimed at mitigating the environmental impact of commercial aircraft. As the number of technology alternatives grows along with model complexity, current methods for Pareto finding and multiobjective optimization quickly become computationally infeasible. Coupled with the large uncertainty in the early stages of design, optimal designs are sought while avoiding the computational burden of excessive function calls when a single design change or technology assumption could alter the results. This motivates the need for a robust and efficient evaluation methodology for quantitative assessment of competing concepts.

This research presents a novel approach that combines Bayesian adaptive sampling with surrogate-based optimization to efficiently place designs near Pareto frontier intersections of competing concepts. Efficiency is increased over sequential multiobjective optimization by focusing computational resources specifically on the location in the design space where optimality shifts between concepts. At the intersection of Pareto frontiers, the selection decisions are most sensitive to preferences place on the objectives, and small perturbations can lead to vastly different final designs. These concepts are incorporated into an evaluation methodology that ultimately reduces the number of failed cases, infeasible designs, and Pareto dominated solutions across all concepts.

A set of algebraic samples along with a truss design problem are presented as canonical examples for the proposed approach. The methodology is applied to the design of ultra-high bypass ratio turbofans to guide NASA's technology development

efforts for future aircraft. Geared-drive and variable geometry bypass nozzle concepts are explored as enablers for increased bypass ratio and potential alternatives over traditional configurations. The method is shown to improve sampling efficiency and provide clusters of feasible designs that motivate a shift towards revolutionary technologies that reduce fuel burn, emissions, and noise on future aircraft.

CHAPTER I

INTRODUCTION

1.1 Background

1.1.1 Engineering Design

Design is a process for describing a physical thing whose existence will fulfill a need or accomplish a goal. Like many processes, there are several basic steps generally accepted within industry and academia to yield good designs. While the names given to each particular phase may differ, the ideas and goals are generally uniform across most disciplines and designs. The first phase of *conceptual design* consists of establishing the need, defining requirements, concept generation and selection. *Embodiment* or *preliminary design* follows where the abstract concept obtained in the previous phase begins to take shape. Parametric studies are performed and structural development of the product takes place. Next is the *detailed design* phase where a complete engineering description of the product is developed with detailed drawings, three-dimensional models, and part descriptions. By the end of the detailed design phase, the physical product is completely described [45, 126, 135]. This does not end the overall design however, as phases less germane to engineering are required. Planning for construction, distribution, implementation and retirement still are performed and usually follow detailed design [17].

1.1.2 Concept Generation and Selection

Concept generation, performed in the conceptual design phase, is the act of creating the design alternatives that can satisfy the given set of requirements. Functional decomposition [146], morphological analysis [194, 195], creative brainstorming, or other methods [96, 156] may be used to generate the often combinatorial space of

alternatives representing all specific arrangements of components and subsystems that make up a final design.

Ultimately, the designer must choose one or small set of concepts that best meet a given set of requirements. Once selection occurs, the concept moves into preliminary and detailed design phases. Concept selection is one of the most important tasks in the design process because it has the most impact on how the final system will look and behave. While small perturbations and growth can occur around the chosen design, the selection of a particular concept dictates what the final system will be and what it can never transform into. In this regard, concept selection is the ultimate constraint placed on a system [40]. This decision has also one of the greatest impacts on overall cost [170, 184].

At this point it is important to be more specific about the term "concept" to avoid ambiguity throughout this thesis. While cursory literature review reveals that no two definitions are the same among even the most seminal sources, some common themes are evident. Namely, a concept is a particular arrangement of components that are brought together physically to fulfill a set of requirements. For each concept there is a direct mapping between functional requirements and the means to perform each function [3, 4, 146]. For example, consider the requirement of providing lift to an aircraft. The function of lift can be performed by a variety of components: rigid-wing (civil transport), rotor (helicopter), vectored thrust (VTOL fighter), lifting gas (blimp), etc. Utilization of a particular method of lift constitutes a single concept.

This thesis will adopt the above definition of a concept as well as add to it in the following way; the arrangement of components must be describable by an executable model that can predict the concept's performance over a range of unique inputs. Performance of each concept can be evaluated through execution of the derived model. This addition to the notion of a concept brings with it a subtle but important fact. Each concept in the set of potential solutions to a design problem has a unique

model with which it is represented. These models are independent with their own sets of inputs (design parameters), but must be measured by common figures of merit/requirements. Consider two concepts from the above example, a rigid-wing civil transport and a blimp. Both the airplane and blimp can fulfill the need to transport a payload through the air, but lift is achieved in two distinct ways. Engineers working on each concept have vastly different sets of degrees of freedoms associated with their designs. A degree of freedom is the designer's ability to control a particular aspect of a design. An engineer designing a rigid-wing aircraft can change aspect ratio, wing thicknesses, chord lengths, and airfoil geometries while the blimp designer may want to adjust envelope material, diameter, length, and ballonet volume to optimize performance. Yet both concepts can be compared to each other along similar metrics: mission range, operating cost, emissions, etc. It is the job of the designer to look at each concept along similar dimensions and choose which to pursue for further development. It is the act of down-selecting from many concepts to a single concept or set of concepts for further investigation that constitutes concept selection.

1.1.3 Multiobjective Decision Making

For single objective problems, there is a natural order to all possible designs and selection of a single best design is straightforward. For many real engineering problems however, the decision maker must consider multiple competing objectives. Competition arises through incommensurability of objectives in real-world systems bound by physical laws. A structural engineer often wishes to reduce cost at the same time as minimizing strain and deflection of a structure under loads. However, the added material required to make a structure more stiff drives up cost.

For the multiobjective problem, the optimization task involves identification of those designs that are *Pareto optimal*. A design is said to be Pareto optimal if none of the objectives can be improved without degrading in at least one other objective

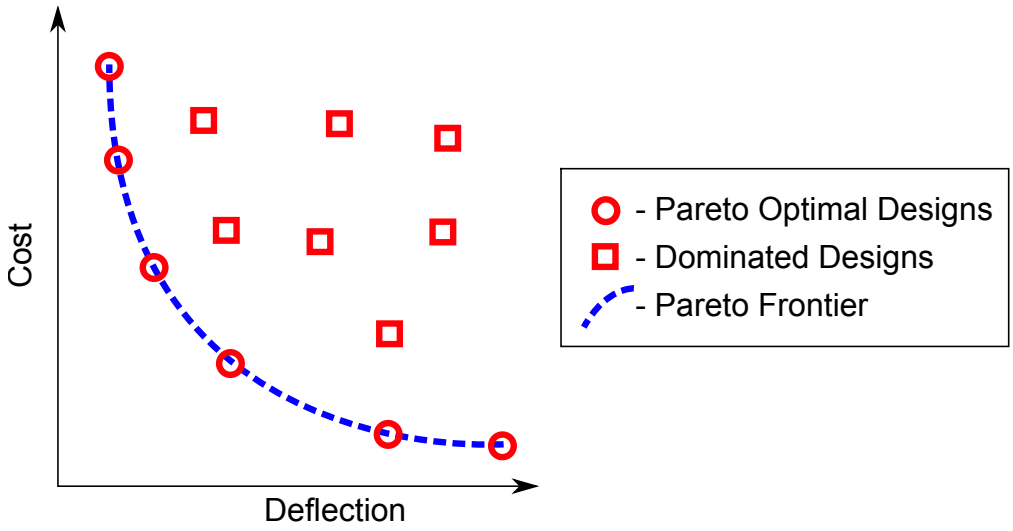


Figure 1: The Multiobjective Problem

[94]. There may exist many or even an infinite number of points that make up a Pareto optimal set where each point represents a specific setting of the design parameters. Rather than locate the potentially infinite number of optimal designs, a theoretical boundary of Pareto optimality can be drawn called a Pareto frontier. The Pareto frontier is the boundary between physical impossibility and realizable performance of a system. All points that fall on the frontier are Pareto optimal. The Pareto set of designs cannot be ordered in the same way as the single objective problem because mathematically, every point in the set is an equivalently acceptable design solution [110]. The notion of Pareto optimality is shown graphically in Figure 1 and continues the example of minimizing cost and deflection in a structure. The two designs on the edges of the blue dotted line are considered "best-in-class" solutions (lowest cost and lowest deflection) and would be obtained with single objective optimization. As design move along the Pareto frontier, this represented trading off between the two objectives. Without specifying preferences on the objectives (e.g. deflection more important), all of the red circles of Figure 1 are equally desirable solutions.

Table 1: Emission and Noise Reduction Goals

Entry into Service	Fuel Burn Reduction	LTO NOx	Cumulative Noise
2015	-33%	-60%	-32 dB
2020	-40%	-75%	-42 dB

1.2 NASA Fundamental Aeronautics Program

According to studies by the FAA and NASA, air transportation demand is expected to double over the next 20 years with domestic enplanements expected to increase between 2.4% and 2.6% per year over that time [5, 6, 9]. The growth of commercial aviation brings to the forefront important environmental concerns as emissions and noise from aircraft will also increase. In order to address these issues, NASA established a Fundamental Aeronautics Program (FAP) which set ambitious goals for reducing fuel burn, greenhouse gases, and noise from aircraft. These specific goals and their targets are detailed in Table 1. The most important environmental metrics are fuel burn, oxides of nitrogen during landing and takeoff (LTO NOx), and cumulative noise. LTO NOx reduction goals are relative to the Committee on Aviation Environmental Protection's (CAEP/6) stringencies and typically measured in grams per kiloNewton. Cumulative noise is the summation of noise measurements in decibels at three locations near the runway: sideline, approach, and takeoff (see [2] for more details on noise certification). The noise metric adopted by the FAP is reduction relative to current Stage 4 certification levels. In addition to these environmental goals, performance metrics including minimum weight and takeoff field length are also desired.

1.3 Ultra-high Bypass Ratio Turbofans

NASA FAP's primary focus is development of technology and advanced concepts to meet these aggressive goals. The Subsonic Fixed-Wing (SFW) research project within FAP has identified the ultra-high bypass ratio (UHB) turbofan engine as a potential

method for reducing noise and emissions of future aircraft [54]. Increasing the bypass ratio (BPR), or amount of air bypassing combustion, improves propulsive efficiency by accelerating a large volume of air while maintaining low pressure ratios across the fan [192]. In addition to lower fuel burn, the lower jet exhaust velocity from lower pressure ratios leads to lower noise.

Bypass flow cannot be increased indefinitely however, as there are significant drawbacks which have capped BPR on modern aircraft at around 11 (optimal BPR of UHB engines is expected to be between 12 and 14). First, there are weight and drag penalties associated with the larger diameter fans as well an airframe integration issues with wider nacelles. Lower fan pressure ratios also lead to fan surge problems and large variation in performance between sea-level and cruise flight conditions. Perhaps the most significant disadvantage to UHB technology is the shaft speed mismatch between low-pressure turbine (LPT) and fan. These drawbacks will be discussed in detail in the following sections along with two specific technologies identified by the SFW project as potential enablers to UHB engines.

1.3.1 Variable Area Bypass Nozzle

Compressor surge (or stall) is an unsteady aerodynamic phenomenon characterized by axisymmetric oscillation in mass flow axially across a stage. In extreme cases, the oscillations are such that flow can reverse directions. This causes performance degradation, structural vibration, and potentially significant damage to the turbine engine [39, 62]. For this reason, fans and compressors are designed with enough "surge margin" to allow safe operation across all operating conditions. Imposing a surge margin, while ensuring safety at low throttle and sea-level conditions, generally leads to degraded performance in cruise portions of the flight envelope. This is shown by the dotted line (operating line) in Figure 2. Fan and compressor performance is often shown in plots of corrected mass flow versus pressure ratio. The surge/stall

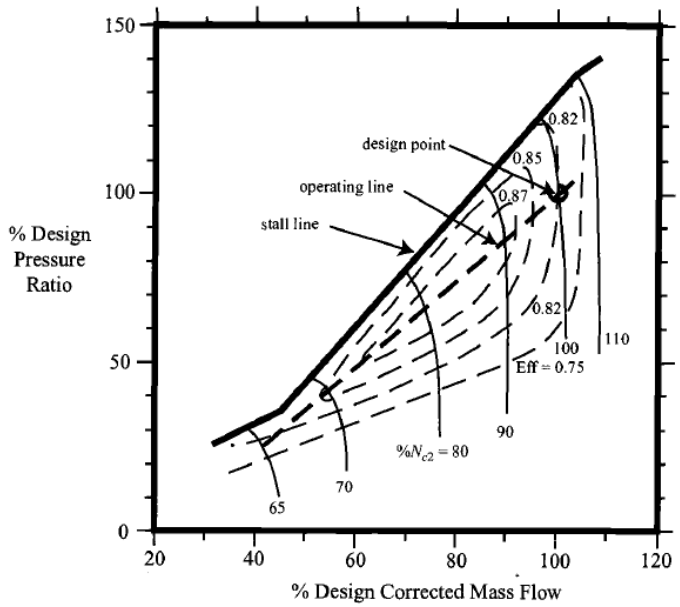


Figure 2: Example Compressor Performance Map [100]

line represents the limit of normal operation of a compressor. Notice how the design point is selected to provide sufficient margin at low pressure ratios which sacrifices efficiency. A *variable geometry bypass nozzle* can alleviate the surge problem by increasing flow area when operating near sea-level to move the fan operating line away from potential surge for low mass flow conditions [109]. There is a tradeoff however; surge is eliminated at the expense of extra weight and component complexity of actuation and control of a variable area nozzle.

1.3.2 Gear-Driven Fan

In general, the fan operates at peak efficiency when operating at low tip speeds in order to maintain low blade Mach number. The LPT must operate at high rotational speeds to reduce loading for a given number of stages. This creates the classical shaft speed mismatch between LPT and fan. Rather than increase LPT diameter which would obstruct the bypass stream, in conventional engines the stage count is often increased which adds weight and length to the engine [23]. As the fan speed and pressure ratio drop in UHB engines, the fuel efficiency gains are countered by severe

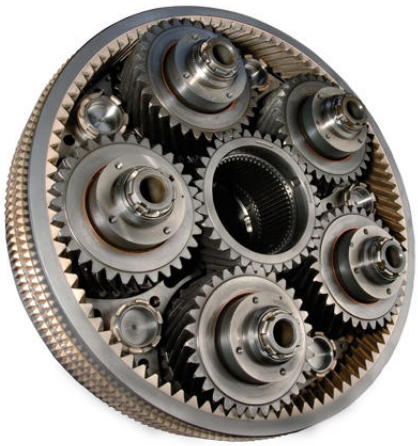


Figure 3: Planetary Gear Component
(2011, http://www.popsci.com/bown/2009/gallery/2009-11/pratt-and-whitney-purepower-pw1000g)

weight penalty associated with adding LPT stages in order to maintain performance. Introduction of a gear between these two components like that shown in the planetary system in Figure 3 allows for reduction of rotational speed for efficient fan operation at the same time affording the flexibility in LPT design in terms of staging [98]. Again, there is a tradeoff, since now the designer must deal with the added weight and reliability issues of a gearbox operating in an environment of extreme rotational speeds and loads.

1.3.3 Historical Perspective

While the idea of ultra-high bypass has existed since the 1970's, the technology has very little practical implementation and yet to be seen in production. Part of the reason for this is that high bypass technology is primarily dependent on the evolution of the core engine, which is approaching its technological limit [155]. High propulsive efficiencies need small cores to produce enough power to drive the fan without increasing airflow. This is done with high pressure ratios and high turbine inlet temperatures. As pressures and temperatures increase inside the engine, tip clearances, materials properties, and cooling become very important [93]. The manufacturing processes and materials that enable these aspects of design have typically lagged the

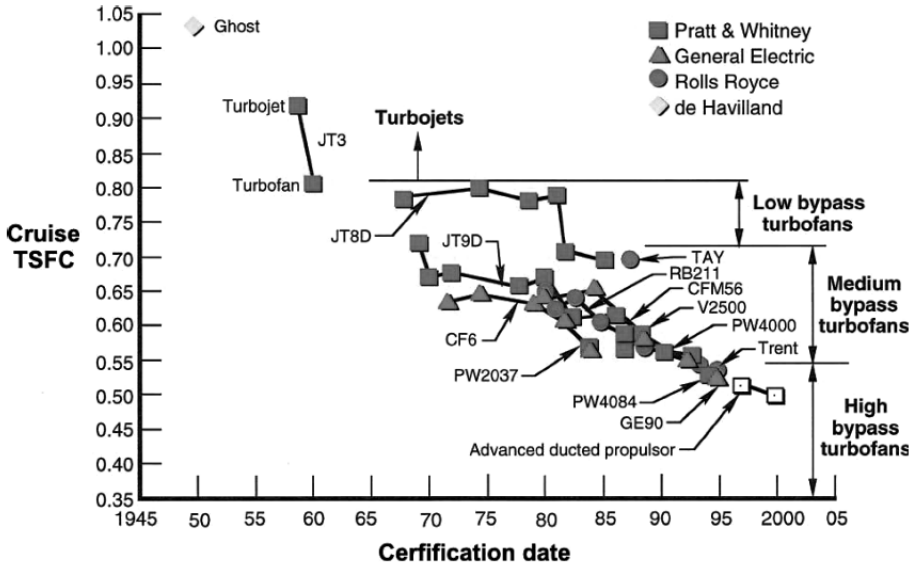


Figure 4: Evolution of Turbine Engine Performance [93]

development of the colder section components. See Figure 4 for the evolution of turbine engine performance.

Pratt and Whitney has pursued geared turbofan technology since the late 1990's and emerged as an industry leader with the PurePower PW1000G scheduled for entry into service by 2013. The PW1000G is in the 30,000 pound thrust class and expected to power the Mitsubishi Regional Jet and Bombardier CSeries. Geared turbofan development has largely been assigned to smaller regional jets where engines can be mounted on the fuselage to avoid installation issues of an under-wing mount. However, rising fuel costs and environmental concerns have driven focus to applications where technology improvements will have a larger impact within commercial aviation. This includes single-aisle, 150 passenger class (e.g. next generation Boeing 737 or Airbus A320) vehicles that are expected to make up 65% of new aircraft produced over the next 20 years [66].

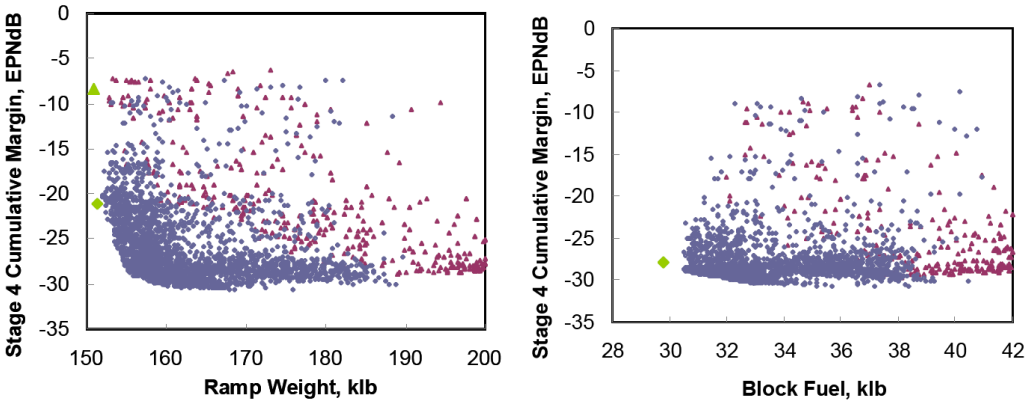
1.4 Benchmarking the UHB Design Problem

Because of the potential reduction in fuel burn and noise offered by UHB engines, several studies have been conducted over the last several decades to investigate their application to wide-body aircraft. In 1972, a general discussion of the benefits and reliability of gear driven fans was presented by engineers at General Motors [98]. A more theoretical investigation into propulsive efficiency and fuel burn was done at the Boeing Company in 1987 [47]. In 1989, Zimbrick and Colehour presented an analytical discussion of thrust specific fuel consumption (TSFC) reduction with high bypass ratio engines [192]. While these studies mostly considered only engine design and rarely analyzed the performance of the entire aircraft, the general conclusion was that fuel burn benefits could not overcome the severe weight penalty of larger fan and gearbox.

Recent increases in fuel prices and desire for noise and emission reduction has refocused efforts on UHB technology. In 2009, Guynn et al. at NASA conducted analysis of overall system performance of gear-driven fan technology on a next generation CFM56-7B turbofan engine and Boeing 737-800 aircraft [66]. The work represented a collaboration between NASA, Army Research Laboratory, and Georgia Institute of Technology where multidisciplinary tools were linked together to form a systems-level analysis. However, the study only considered one objective at a time as each varied with fan pressure ratio (FPR). Naturally, for the single objective problems, optimal values of fuel burn, noise, and emissions were achieved for different values of FPR. To address the multiobjective nature of the problem, Berton and Guynn perform several constrained bi-objective optimizations [23]. The analysis considered constraints such as takeoff ground roll, approach velocity, and rate of climb among others. While analysis only considered two objectives at a time and required a significant amount of computation time (tens of thousands of function calls representing several weeks on a single-core computer), some valuable insight into the technology tradeoff was

Table 2: Single Objective UHB Engine Results [23]

Single Objective Solution	Concept	FPR
Min Ramp Weight	Direct Drive Fan	1.70
Min Block Fuel	Gear Driven Fan	1.36
Min Noise	Gear Driven Fan	1.35
Min LTO NO_x	Gear Driven Fan	1.35



(a) Weight/Noise Trade (b) Block Fuel/Noise Trade

Figure 5: Sample Results for Biobjective Optimization [23]

discovered. Single objective results along with the corresponding optimal FPR are summarized in Table 2.

The main conclusion to draw from Table 2 is the shift in preference from direct drive to gear driven technology when considering fuel efficiency and environmental metrics. While the data only reflect single objective optimization, it is clear that a tradeoff exists between gear-driven and direct-drive fan technology. In order to better characterize this tradeoff, multiobjective optimization was performed, and sample results are shown in Figure 5. The geared fan architecture is shown by blue diamonds, and direct-drive are magenta triangles. The single objective optima from Table 2 are plotted along with the multiobjective optimization results and shown in green.

1.4.1 Limitations

While the minimum ramp weight solution appears to favor direct-drive technology obtained from single objective optimization, it is still unknown where the shift in

technology occurs. This introduces an important limitation with current concept evaluation methods and provides strong motivation for this research. Traditional analysis methods such as evolutionary optimization used in [23], are highly susceptible to the situation where *relative* performance between concepts is modeled quite poorly. This leads to little understanding of the relationship between concepts in the area where Pareto optimality shifts from one concept to another. From a concept selection standpoint, this is the area that is most important to the designer because here the decisions are most sensitive to preferences placed on the objectives. Small perturbations in preference can lead to vastly different designs which have large implications on overall cost.

Another limitation is highlighted in Figure 5a by the dense sampling of points in suboptimal areas for both concepts. Most of the computation time was devoted to exploring these suboptimal regions rather than Pareto optimal solutions that inform selection decisions. The problem is now summarized considering these limitations in the following section and roadmap for further research is laid out.

1.5 Problem Summary and Goals

As vehicle requirements become more aggressive, engine manufacturers are forced to pursue concepts beyond conventional configurations. This pushes the designs into areas where engineers have little prior knowledge to meet a combination of environmental and performance goals.

This thesis will assess the performance of two enabling technologies for UHB engines and compare them to conventional direct-drive architecture. The specific focus will be application of geared turbofan and variable area bypass nozzle on a next generation single-aisle transport aircraft (150 passenger class). The conceptual design of these configurations will inform concept selection decisions as well as guide FAP's technology development efforts as NASA seeks to support growth and minimize the

environmental impact of commercial aviation.

1.5.1 Challenges

1.5.1.1 Constrained, Multiobjective Decision Making

At its core, pursuance of UHB engine technology is a multiobjective decision making (MODM) problem. Emissions, noise, fuel burn, and weight must all be minimized under several constraints in order to meet NASA's aggressive goals for next-generation aircraft. The final concept ultimately represents a tradeoff between all of these objectives and constraints.

At this stage in conceptual design, the tradeoff in performance of a single concept is less important than placing that tradeoff in the context of other concepts. The problem is not only to evaluate each concept, but to *synthesize* all of the information to support design decisions. This research aims to not only evaluate the tradeoffs between objectives of UHB enabling technologies independently, but to characterize those tradeoffs that may warrant selection of the technology over traditional engine designs.

1.5.1.2 Implications of Concept Selection

In the early stages of design when most concept and technology decisions must be made, very little detailed information about the shape and behavior of the final system is available. If the designs are evolutionary improvements over previous technology, this is less of a concern as experienced designers know approximately how an optimal design will look. When revolutionary technology advances such as UHB engines are considered, traditional design methods often cannot support conceptual design studies [15]. The large amount of uncertainty in conceptual design of revolutionary systems makes selection a very difficult task.

1.5.1.3 Computational Considerations

NASA's goals for next generation commercial aviation are at the system level. That is, they cannot be assessed by examining the engines independent from the design of the entire aircraft. Furthermore, aircraft/engine performance must be combined with a capability for investigating objectives such as noise and emissions which are not typically available in sizing and synthesis codes. Concept evaluation will require coupled, multidisciplinary tools to evaluate every objective which introduce computational complexity and thus increase the time required for a single design solution. This is the case for the UHB design problem where thermodynamic cycle models, aircraft sizing & synthesis codes, noise approximations, and emissions models must be executed in series for a single design.

Computational cost is even more critical when considering that the models are not static throughout conceptual design. They are in fact constantly evolving as decisions are made. Degrees of freedom are nailed down, experimental data becomes available, and fidelity is increased in different regions of the design space placing a high priority on efficiency. Therefore, too much time searching for *the answer* may be wasted if a small design change, technology assumption, or introduction of a new constraint affect performance predictions.

Computer codes, when used for conceptual design necessarily contain many assumptions to enable automation during optimization or design space exploration. Even with these built-in heuristics, many tools are nevertheless sensitive to convergence issues and constraint violation. Thermodynamic cycle modeling and noise prediction, components to a UHB evaluation, are especially vulnerable to these factors. Failed cases or otherwise infeasible points contribute to the computational expense and must be factored into the total cost of evaluating a single concept.

1.5.2 Research Objectives

The outcome of the research will be a methodology for analysis of competing concepts in the presence of multiple objectives and constraints. The process must be capable of handling many concept alternatives with an arbitrary number of objectives and design variables. The method will be repeatable, transparent and reduce uncertainty in conceptual design through quantitative assessment of concept performance. The method must be capable of assessing revolutionary concepts backed by expensive models and place a high priority on computational efficiency, where each function call is at a premium. To increase efficiency, the method will synthesize information from each concept and focus on reducing uncertainty where concept selection decisions are the most sensitive to preferences on the objectives. In addition, the number of failed cases and infeasible points will be minimized while maintaining automated evaluation.

The methodology will be applied to turbofan concept selection for a single-aisle transport aircraft, expected to make up a large portion of future commercial fleet. New technology on these next-generation aircraft has the potential to greatly reduce emissions, noise, and overall impact on the environment. For this reason, NASA is actively pursuing gear-driven fans and variable geometry nozzle as key enablers to UHB technology.

1.5.3 Research Questions

Two high-level research questions are now posed to further guide this research and support development of a new methodology for concept evaluation and selection.

- **Research Question 1:** What is the state of the art in concept evaluation and selection?

  This question leads to a literature review of the previous work done in the area. In answering this question, gaps between traditional design methods and the

research objectives will be identified to be addressed by the proposed methodology.

- **Research Question 2:** How will the problem of constrained, multiobjective decision making with independent, expensive models be solved?

  The challenges listed in Section 1.5.1 must be addressed by the new methodology in order to solve the UHB engine selection problem. High priority is placed on computational efficiency and comparing/contrasting between concepts rather than just accurate depiction of individual concepts. Designers would like to understand the implications of making concept selection decisions and the risk associated with his/her preferences that lead to selecting one alternative over another.

1.6 Thesis Organization

- Chapter II will present background into both classical methods and the state of the art of concept selection. It will explore the most popular methods within industry and academia that are both qualitative and quantitative in nature.

- Chapter III - Methods will be investigated to address the multiobjective nature of the problem. Additionally, techniques for Pareto finding and optimization of expensive functions will be presented which will partially answer Research Question 2.

- Chapter IV - The research questions will be revisited along with presentation of a top-level hypothesis that will guide the remaining work. Technical details will be provided of some of the building blocks to a new methodology. A formal methodology will be defined that combines elements of surrogate-based optimization, Bayesian adaptive sampling, and multiobjective decision making.

- Chapter V - A set of low-level research questions and hypotheses will be listed

along with computer experiments to investigate the performance of a new evaluation methodology on two canonical problems. First an algebraic sample problem will be studied followed by a common engineering problem of truss design.

- Chapter VI - The formal methodology will be applied to the problem of UHB engine concept selection. A description of the problem, modeling approach and assumptions will be documented. The chapter will conclude with visualization and analysis of the evaluation results.

- Chapter VII - Overall conclusions will be presented and some recommendations for using the new evaluation methodology. The top-level research questions and hypothesis will be revisited and areas for future research will be identified.

CHAPTER II

CONCEPT SELECTION STATE-OF-THE-ART

The importance of concept selection is by no means a new idea in design. Research into a new methodology for concept evaluation and selection cannot progress without first identifying what is currently being used for the task. This chapter will take a more in-depth look how the problem has been and is currently handled. Background in numerical optimization and surrogate modeling is also presented which enable expensive analysis codes in early design.

2.1 Qualitative Concept Selection

There is a class of methods often employed in academia and industry for concept selection that are qualitative in nature. These methods have little or no mathematical basis and typically rely on prior knowledge and designer experience. Some methods discussed in this section attempt to bring some transparency to the decision making process by adding a layer of simple mathematics to a sufficiently decomposed problem.

2.1.1 Group Preferences and Voting

The first group of concept selection methods is purely based on intuition and prior knowledge of a group of decision makers. The success of these methods is strictly dependent on an engineer's ability to predict performance in the early stages of design when very little is known about the problem. Differing opinions among engineering groups, competing design and manufacturing objectives, the presence of unquantifiable objectives, and overall design uncertainty makes the task of negotiation a difficult one [151]. Even in the presence of much uncertainty, concept review meetings are one of the most often utilized concept selection methods [147]. Group preferences

Table 3: Technology Readiness Level

TRL	Description
1	Basic principles observed and reported
2	Technology concept and/or application formulated
3	Analytical experimental critical function and/or characteristic proof-of-concept
4	Component and/or breadboard validation in laboratory environment
5	Component and/or breadboard validation in relevant environment
6	System/subsystem model or prototype demonstration in a relevant environment
7	System prototype demonstration in a target/space environment
8	Actual system completed and "flight qualified" through test and demonstration
9	Actual system "flight proven" through successful mission operations

are often imposed by voting schemes or open discussion of each competing concept. Methods such as Go-No-Go or feasibility screening [171] as well as assessment of Technology Readiness Level (TRL) [4] serve as the first step in a multi-part, iterative selection method. TRL was introduced to provide an easy way to recognize the risk associated with pursuing a given technology (see Table 3). These methods and others, while lacking in quantitative assessment capability, are very capable of ruling out many competing concepts and reducing the combinatorial space of solutions to a more manageable subset of alternatives.

These methods however, lack traceability. In making a decision to pursue one concept over another, an engineer often appeals to his/her intuition or 'gut feeling'. In addition to traceability issues, there are other significant disadvantages to voting and group preference approaches. Primarily, there is no obvious way to address differing preferences among a group of decision makers [165]. Differing opinions are combined arbitrarily depending on various engineers' perceived experience with a concept or position within an organization. There is also a limit to an engineer's ability to gather and process information [111]. This problem becomes more pronounced as the number of alternatives grows. Revolutionary concepts further diminish a designer's

Table 4: Typical Rating Scheme for Weighted Decision Matrices

Rating	Description
1	Poor performance along objective
2	Fair performance along objective
3	Average performance along objective
4	Good performance along objective
5	Excellent performance along objective

ability to predict behavior because there is little prior knowledge upon which to call. Methods often employed to avoid these limitations will be discussed in the following sections.

2.1.2 Decision Matrices

Decision matrices introduce some traceability to the selection process by breaking the problem down into simple decisions for each alternative. Concepts are rated along each objective one at a time, then the ratings are combined algebraically into a single objective function. The ratings are still based on engineer input, but the decomposition brings more transparency to the final decision of which concept to pursue. This section will investigate common methods for assigning ratings and how those scores are typically combined to a single number engineers wish to maximize.

2.1.2.1 Weighted Decision Matrices

One of the most popular forms of decision matrix is the weighted decision matrix (WDM). Each concept is assigned a rating according to its expected performance in that objective. A common ratings scheme is provided in Table 4. Once the concept has been rated along each objective, the total score is obtained using Equation 1 and 2. In addition to the individual ratings, weights are placed on each objective to signify the importance of that objective to the decision maker. Higher weight indicates that a larger percentage of the total score is attributable to performance along that objective.

Table 5: Weighted Decision Matrix

	Weight	Concept 1	Concept 2	Concept 3	Concept 4
Objective 1	0.7	1	5	2	1
Objective 2	0.3	3	3	4	4
Total Score	–	1.6	4.4	2.6	1.9

$$S^i = \sum_{j=1}^{n} w_j \mu_j^i \tag{1}$$

$$\text{Subject to:} \quad \sum_{j=1}^{n} w_j = 1 \tag{2}$$

where S^i is the total score for the i^{th} concept, n is the number of objectives, w_j is the weighting of the j^{th} objective, and μ_j^i is the rating of the i^{th} concept along the j^{th} objective.

A sample weighted decision matrix for four concepts and two objectives is shown in Table 5 using the rating scheme given in Table 4. In this example, performance along Objective 1 is more important to the designer as evidence from the weightings on each objective. Concept 2 is the most favorable given these weightings because it has the highest total score.

In addition to the tabular form seen in Table 5, there is also a graphical interpretation for problems of two objectives. When plotted as a function of two objectives, each concept can be represented as a single point. An example is shown for two objectives and three concepts in Figure 6. This figure also illustrates a major disadvantage of weighted decision matrices: the inability to account for solutions that lie on non-convex regions of Pareto optimality [116]. The dashed lines in the figure are lines of constant total score. In this example, Concept 3 is a non-dominated solution representing a tradeoff between Objective 1 and Objective 2, but the method utilizing Equation 1 will never offer up Concept 3 as desirable alternative. Some attempts have been identified in the literature to overcome this inability to capture non-convex Pareto solutions. These methods, called compromise programming or weighted square sum method rely on slight modifications to the overall objective function shown in

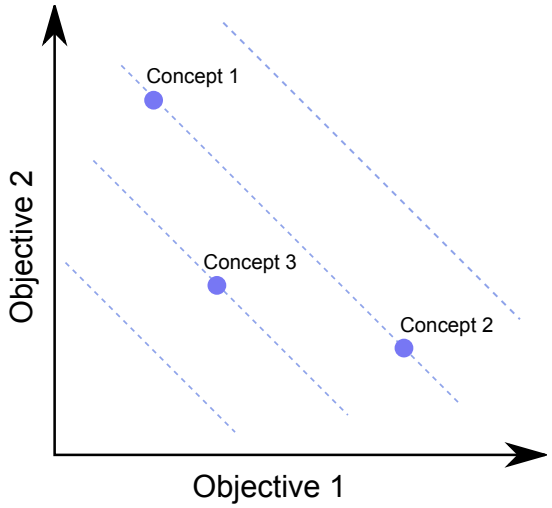


Figure 6: Graphical Interpretation of Weighted Decision Matrices

Equations 3 and 4 [106]. In addition to compromise programming, Athan and Papalambros explored the most general functional forms of the overall objective with additional tunable parameters for capturing all possible Pareto points [18].

$$S^i = \sum_{j=1}^{n} w_j(\mu_j^i)^m \tag{3}$$

$$S^i = \sum_{j=1}^{n} w_j(\mu_j^i - \mu_j^*)^m \tag{4}$$

Where m is an even integer greater than or equal to 2 and μ_j^* is a utopia point for j^{th} objective.

2.1.2.2 Pugh

The Pugh evaluation matrix is modification to the standard weighted decision matrix and avoids convexity issues by forcing all objective weights to be equal to one. Additionally, the score for a single objective can be either be +1 (better than), 0 (the same as), or -1 (worse than) in reference to a baseline concept [129]. An example Pugh matrix for four concepts and three objectives is shown in Table 6. The graphical interpretation is shown in Figure 7. Adopting this weighting and ranking

Table 6: Pugh Evaluation Matrix

	Baseline	Concept 2	Concept 3	Concept 4
Objective 1	–	0	-1	+1
Objective 2	–	0	-1	+1
Objective 3	–	+1	+1	0
Total Score	–	+1	-1	+2

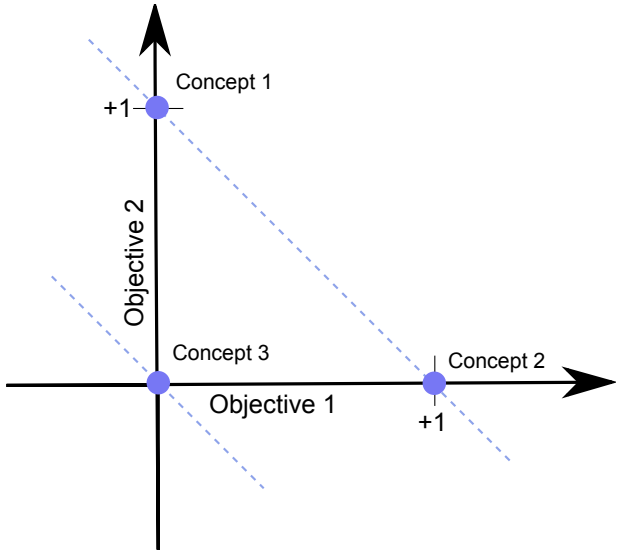


Figure 7: Pareto Frontier Formed from Pugh Evaluation Method

scheme inhibits formation of concave Pareto fronts because the only attainable scores are those that are dominated by other points.

Although Pugh addresses the issue of capturing points in non-convex regions, he does so through a loss of information; all the objectives are equally important. The inability to place preferences on objectives is seen as a major drawback to using Pugh evaluation matrices for concept selection. In general however, decision matrix methods do not address the difficulty in resolving conflicting opinions among multiple decision makers. Pairwise comparison methods, discussed in the next section, specifically address this issue.

2.1.3 Pairwise Comparison

The basic idea behind pairwise comparison methods is to perform $\frac{1}{2}(n^2 - n)$ separate evaluations where n is the number of concepts. In each evaluation, a pair of concepts is compared and ranked on all objectives at once. Once this is performed for all pairs, the result is an ordered list of concepts for each decision maker. The task then becomes combining the different lists into a single measure to obtain the most preferred concept. However, combining differing opinions leads to two common problems of pairwise comparison methods [142]. First, there is a susceptibility to rank reversal where preference of one concept over another changes with the addition or subtraction of a third concept. Second is the possibility to get non-intuitive results where an alternative is selected that loses all pairwise comparisons. Two methods discussed in [48] are designed to overcome these limitations: Borda Counts and Pairwise Comparison Charts (PCC). Both use a rating scheme where a point value is assigned to each rank then the number of times each concept appears in each rank is tallied. This produces a total score that is maximum for the most preferred concept. Consider the example where 30 designers are asked to rank five design concepts resulting in the following ordered lists.

- 10 Concept 1$\succ$Concept 2$\succ$Concept 3$\succ$Concept 4$\succ$Concept 5
- 10 Concept 2$\succ$Concept 3$\succ$Concept 4$\succ$Concept 5$\succ$Concept 1
- 10 Concept 3$\succ$Concept 4$\succ$Concept 5$\succ$Concept 1$\succ$Concept 2

where $x \succ y$ indicates that x is preferred over y.

Each time a concept is ranked first, it receives four points. Second place receives three points, third place receives two points and so on. The resulting Borda chart and PCC are shown in Tables 7 and 8. In both methods, Concept C is the most preferred. If one concept is removed and the analysis ranking performed again, the order is preserved.

Table 7: Sample Borda Count for Five Concepts

	Points	*Sum*
Concept 1	40+0+10	50
Concept 2	30+40+0	70
Concept 3	20+30+40	90
Concept 4	10+20+30	60
Concept 5	0+10+20	30

Table 8: Sample Pairwise Comparison Chart for Five Concepts

Win/Lose	Concept 1	Concept 2	Concept 3	Concept 4	Concept 5	*Sum/Win*
Concept 1	–	10+0+10	10+0+0	10+0+0	10+0+0	50
Concept 2	0+10+0	–	10+10+0	10+10+0	10+10+0	70
Concept 3	0+10+10	0+0+10	–	10+10+10	10+10+10	90
Concept 4	0+10+10	0+0+10	0+0+0	–	10+10+10	60
Concept 5	0+10+10	0+0+10	0+0+0	0+0+0	–	30
Sum/Lose	70	50	30	60	90	–

2.1.3.1 Analytical Hierarchy Process

A method for selecting concepts when the objectives are hierarchical in nature is the Analytical Hierarchy Process (AHP), widely used within in industry and academia [190]. Developed by Thomas Saaty in the 1970s, AHP relies on decomposition of the top-level problem into lower level criteria [143]. Then competing alternatives are compared against each other according to the various sub-criteria. A sample problem decomposition is shown in Figure 8. Transparency is increased over Borda and Pairwise Comparison Charts through decomposition into simpler comparisons based on individual objectives rather than overall performance. In addition to simple better or worse relationships seen in previous methods, AHP allows for specification of how much better one concept is over another relative to a given objective according to Saaty's Fundamental Scale [144]. This is shown in Table 9.

AHP is easy to understand and implement, and that has established it as one of the most widely used methods for concept selection for real design problems. The

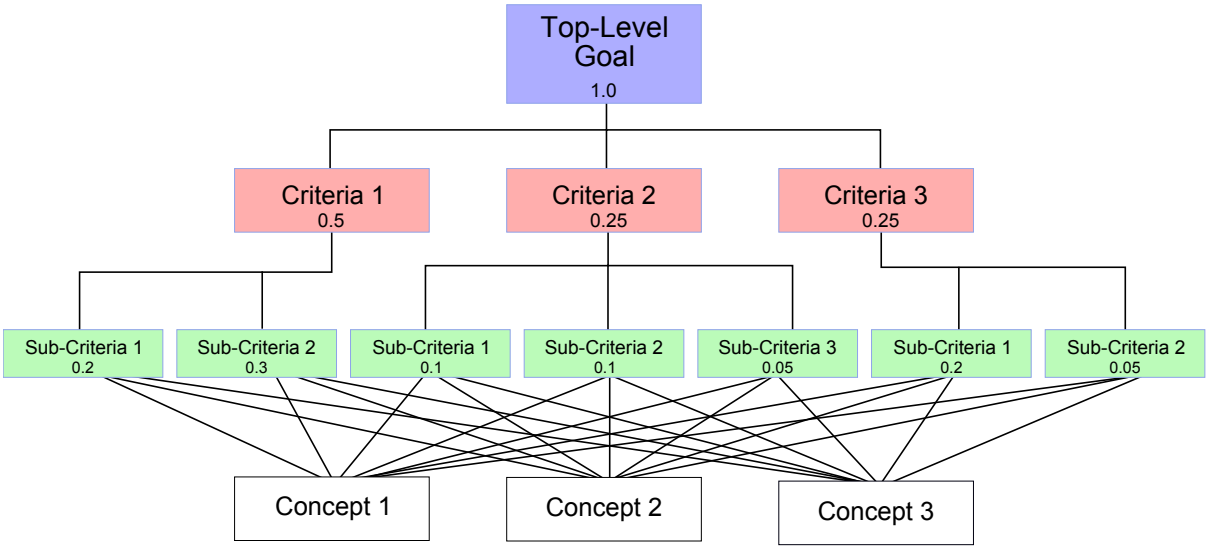


Figure 8: Example Decomposition for Analytical Hierarchy Process

Table 9: Saaty's Fundamental Scale for Pairwise Comparison

Importance	Definition	Explanation
1	Equal Importance	Two activities contribute equally to the objective
3	Moderate Importance	Experience and judgment slightly favor one activity over another
5	Strong Importance	Experience and judgment strongly favor one activity over another
7	Very Strong Importance	An activity is favored very strongly over another and demonstrated in practice
9	Extreme Importance	The evidence favoring one activity over another is the highest possible order of affirmation

method still suffers from rank reversal phenomenon and does not address uncertainty of concept performance in the early stages of design. AHP is also incapable of ruling out concepts or groups of concepts that violate constraints or are otherwise unacceptable [14]. Furthermore, the number of individual comparisons can grow intractably large for a large number of competing concepts. Some methods that have been developed to address uncertainty issues include Fuzzy AHP [188], Fuzzy Set Theory [14, 180], probabilistic decision making [24], and Portfolio Selection Theory [158, 176]. In general, these methods treat uncertain variables as ranges rather than discrete points. Some non-pairwise comparison methods rooted in decision theory have been developed to decrease the amount of work required for concept selection including Technique for Order Preference by Similarity to Ideal Solution (TOPSIS) [80] and Hypothetical Equivalents and Inequivalents Method (HEIM) [154]. They aim to establish objective weights and rank alternatives based on simple questions of preferences posed to the decision maker.

2.1.4 Summary

While there exist many qualitative methods for concept selection beyond what was examined in this section, the most popular are highlighted here. Ordinal methods such as Borda, Pairwise Comparison Charts, and the Pugh Matrix are the simplest to implement and provide a ranking of concepts without numerical values assigned to the objectives. If the decision maker would like to capture *how much* better one concept is over another, numerical values can be assigned using arbitrary measurement scales or actual performance along each objective [124]. This is the approach taken with weighted decision matrices.

In general, the purely qualitative selection methods such as voting and screening reduce the problem to easily understood statements of preference. They are simple, systematic, and easy to use [88]. The decision matrix-based methods even go a step

further to bring some level of transparency to a decision making process where many feel complex mathematics only get in the way. Engineering intuition and tacit knowledge can be valuable sources of insight into product performance as they can provide faster solutions than more sophisticated design methods. Because of high risk for a company to pursue revolutionary concepts or ideas, many designs are evolutionary or incremental changes over well-established designs [61]. This makes relying on engineer experience even more justifiable. This is the main reason these methods are preferred and widely seen in real design problems [50].

For all of the advantages of the more qualitative selection methods, there are some glaring deficiencies that cannot be overlooked. These methods have tenuous, if any mathematical basis. Results are often not repeatable as with a more rigorous approach. Another and perhaps more fundamental flaw with all implementations of pairwise comparison and voting procedures is violation of Kenneth J. Arrow's Impossibility Theorem, or simply Arrow's Theorem [74]. In [16], Arrow proved that fair voting cannot occur when aggregating more than two alternatives. Scott and Antonsson [153] make the distinction between multiobjective decision making in engineering and social choice, and they argue that Arrow's Theorem only has indirect consequences in design, if any. By placing weights on objectives and understanding the interactions of those objectives, designers can circumvent Arrow's Theorem and ultimately select the best alternative without unexpected results. However, care must be taken to explicitly define preferences and avoid arbitrary aggregation of those preferences into a single quantifiable metric [152].

If potential violation of Arrow's Theorem is not a deal breaker for use of qualitative methods in concept selection, the next idea is; each concept is represented by only a single point (Figures 6 and 7). A common theme among these methods is that they ignore the potential for each concept to be a parameterization of its own design variables. The entire Pareto frontier of potential growth for a single concept is not

captured. At best, preferences are specified *a priori* and the single representative concept is ideally (though not always) a reflection of those preferences. If preferences change, the optimal design will change. Also ignored is the fact that preferences are often so poorly understood in early design that the engineer would like to know how they affect the final decision before committing to a single solution [161]. Furthermore, as concepts become more revolutionary to meet a growing number of new requirements, they move into less well understood regions of objective space. Concept complexity can easily increase beyond what an engineer can intuitively grasp and rate accurately. This motivates more numerical-based methods for concept evaluation and selection.

2.2 Quantitative Concept Evaluation and Selection

In this and subsequent sections, focus is turned to a more preferred (if less utilized) class of methods that is fundamentally more rigorous and based in quantitative analysis [147]. These methods have a mathematical basis where objectives are assumed to be scalar quantities and computer models are executed to predict concept performance and inform decisions [73].

Some practitioners argue that using deterministic, physics-based models for concept selection hinders early decision making by overlooking the inherent uncertainty and poorly defined nature of systems in early design [189]. Some feel anything but a high-level, qualitative approach is inappropriate for the task [130]. While qualitative or other 'reductionist' methods for concept selection cannot be ignored, many of these techniques fall short for reasons discussed in the previous section. The migration towards more mathematically rooted, quantitative concept selection is driven mainly by one overarching reason; there is a key desire in the design community for more effective use of modeling and simulation (M&S) early in the design process [65, 77, 178]. This allows disciplinary experts to remain confident that the latest

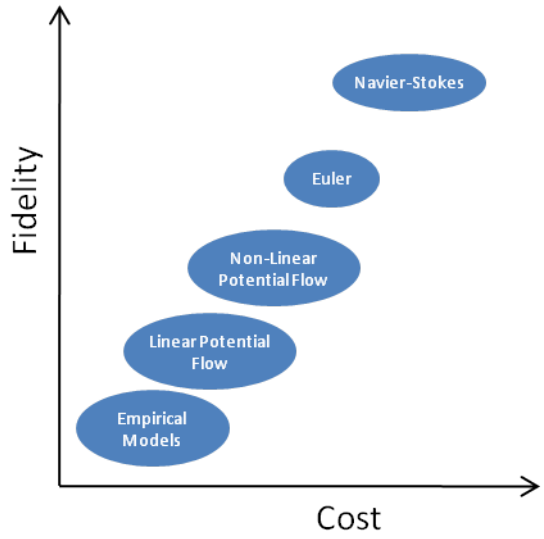


Figure 9: Fidelity and Cost Tradeoff for Aerodynamics Analysis

and most accurate simulation results inform the design decisions. For many modeling tasks, there is tradeoff between accuracy and computational expense associated with time-consuming model executions [163]. An example of this tradeoff within the aerodynamics discipline is depicted in Figure 9 [131]. When concepts are backed by expensive models, it can easily become infeasible to perform a single analysis, much less investigate the objective space of multiple competing concepts. Many conceptual designers have identified this gap between available and required computational resources and recognize it as one of the key difficulties in early design [21, 159].

One way this difficulty can arise is through use of high fidelity modeling tools. Concept performance prediction often depends on capturing complex physics that low-fidelity tools or empirical models are incapable of handling [55, 125]. An example of such a scenario is shown in Figure 10 [131]. If design requirements lead to revolutionary concepts, there are no historical trends or empiricism with which to predict system behavior [30, 132]. At first glance, the modeling effort alone seems to dominate the amount of work necessary for good concept selection. It may even be paradoxical to spend much effort developing high-fidelity tools if it will only be used to establish

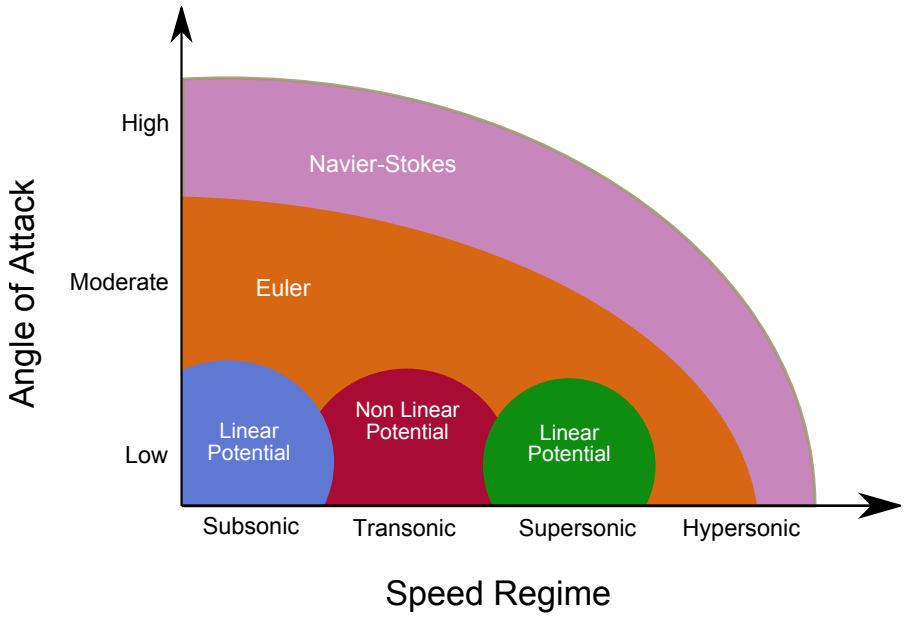


Figure 10: Applicable Regions for Various Aerodynamics Analysis Methods

that a concept is bad. Bridging this gap between concept generation and modeling is the focus of much ongoing research, mainly in the fields of artificial intelligence and creativity [27, 187] as well as evolutionary algorithms (EAs) [64, 137, 138, 140]. Even some conceptual design tools such as Vehicle Sketch Pad (VSP) are seeing more widespread use in order to capture complex physics of revolutionary systems. VSP is a parametric geometry modeling tool developed at NASA Langley Research Center designed to quickly generate complex shapes and configurations [67]. The geometries can seamlessly integrate into CFD or finite element packages providing a high degree of fidelity for revolutionary systems with little modeling effort [133].

The focus does not just have to be on high-fidelity tools to encounter the high computational cost of M&S. Low fidelity codes, while relatively simple on their own, can be linked together into a single *meta*-analysis. This leads to more complex simulations that can become bogged down to the point that evaluation of a single concept is very expensive. Customer requirements often dictate that a wide range of interdisciplinary modeling tools be used for a single analysis. When the designer strives to meet performance, economic, and environmental goals concurrently, closely coupled

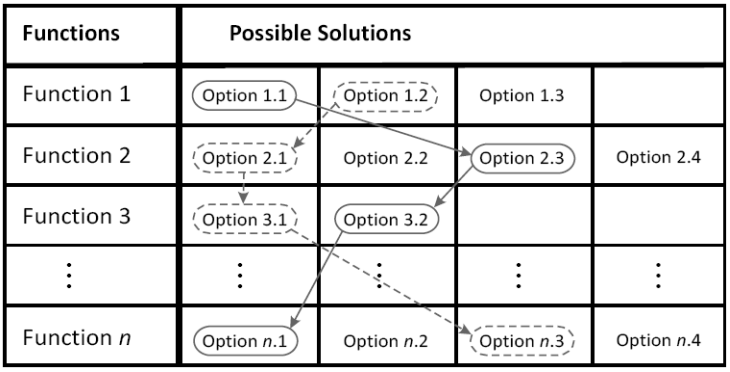


Figure 11: Concept Definition Using a Morphological Matrix [20]

model executions within larger frameworks are a necessity. Feedback loops, interdisciplinary compatibility, internal sub-optimizations and/or sensitivity analyses may be required for a single converged analysis.

2.2.1 Combinatorial Optimization

Early in conceptual design, a functional decomposition of the requirements often leads to a combinatorial set of potential concepts. Product sub-functions are mapped to physical aspects of the design and often presented in graphical form called a morphological matrix [194]. A concept is thus defined as a unique combination of components that satisfy all sub-functions. This is illustrated in the general morphological matrix in Figure 11. A unique concept is a particular path vertically through the matrix with one option selected from each row. Two concepts are shown in the example.

Rather than investigate every possible combination of alternatives by hand, designers can take advantage of the graphical structure of the morphological matrix. Graph theoretic implementation of evolutionary algorithms have been developed that operate on the morphological matrix to identify optimal concepts [20, 31]. Integer programming (IP) and discrete optimization methods can search the combinatorial space of alternatives more efficiently than full-factorial exploration. This has made combinatorial optimization methods particularly well-suited for technology selection and portfolio planning problems [95, 127, 141].

While quantitative measure of concept performance is available, combinatorial optimization methods focus more on handling a large number concepts rather than sufficiently exploring the parametric capabilities of individual concepts. Furthermore, these methods are highly dependent on the ability to *automatically* generate and evaluate concepts. To perform numerical optimization, the components satisfying the subfunctions must be easily combined into a single parameterized model leading to sacrifices in fidelity.

2.2.2 Sequential Multiobjective Optimization for Concept Selection

The focus now shifts towards another class of concept evaluation methods that aim to capture concepts' parametric performance along a Pareto frontier. In identifying those designs that are Pareto optimal, the problem is ultimately transformed into multiobjective optimization.

When posed as a multiobjective optimization problem, an intuitive and natural approach for concept evaluation is independent analysis of each concept. Results are generated for individual concepts in isolation then brought together to make design decisions. Concept models are executed independently allowing optimization tasks to be performed in parallel. Model owners can maintain control of their respective codes and use their expertise to generate Pareto optimal designs in the context of a single concept. An example of this approach is shown in Figure 12. The designer would like to minimize two objectives and must choose between three available concepts. Modeling and simulation has produced sets of Pareto optimal points (black dots) for each concept. Selection decisions can only be made when the resulting Pareto optimal designs are brought together as shown in Figure 13. For this example, Concept C is never preferred over A and B. There exists a tradeoff between A and B where Concept A is preferred if Objective 1 is more important. The numerical optimization techniques will be discussed in a later section, but some applications of this independent

analysis approach are presented here.

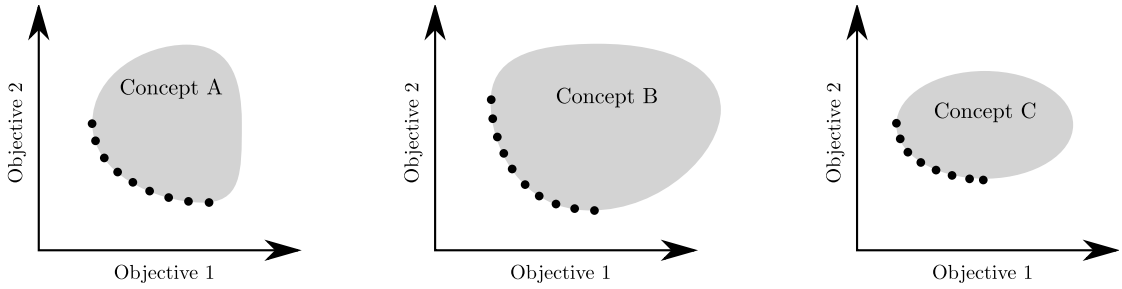


Figure 12: Sequential Optimization of Three Competing Concepts

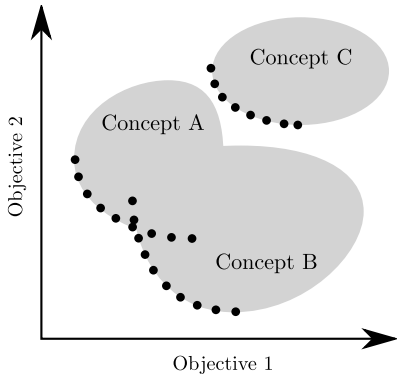


Figure 13: Synthesis of Pareto Optimal Points

Most applications in the literature use Multiobjective Evolutionary Algorithms (MOEA) to generate Pareto frontiers specifically for comparing multiple concepts. Andersson uses multiobjective struggle genetic algorithm (MOSGA) for design of hydraulic actuation systems [13]. Crossley et al. use two-branch tournament genetic algorithm to generate Pareto optimal sets for both truss design [33] and engine selection [34]. An interactive strength Pareto evolution algorithm (SPEA2) is used by Buonanno to find Pareto optimal designs of small supersonic transport concepts [26]. While evolutionary optimization is popular for multiobjective problems, the sequential optimization framework provides the designer flexibility to use any desired technique. More details into how Pareto optimal designs are actually found will be discussed in Chapter 3.

2.2.3 Simultaneous Multiobjective Optimization

The most recent work on quantitative concept selection is in the area of simultaneous optimization. Rather than perform sequential, independent optimizations, this approach seeks to generate the set-Pareto (s-Pareto, concept-Pareto, or c-Pareto in the literature) frontier through simultaneous execution of models [115]. This approach differs from the sequential optimization in that individual Pareto frontiers are never fully formed. Rather, an s-Pareto frontier is built that contains that are Pareto optimal across all concepts. Simultaneous evaluation does not view each concept in isolation but rather considers performance relative to other concepts.

The first notion of s-Pareto frontier appears in [102] where Mattson et al. use a constraint-based approach to obtain optimal designs across all concepts. Avigad and Moshaiov are the first to use an evolutionary approach for establishing the s-Pareto frontier. In [19], a modification to Non-Dominated Sorting Genetic Algorithm (NSGA-II) is proposed to simultaneously evolve populations of non-mating individuals (designs from different concepts).

The fundamental goal of these set-based approaches is to avoid evaluating designs that are dominated and locate designs that are s-Pareto optimal, regardless of concept. This situation is depicted in Figure 14 for the three-concept example shown previously. With this approach, the designer is most concerned with the s-Pareto frontier: optimal designs from Concept A and B and none from Concept C.

2.2.4 Summary

The methods for quantitative concept selection discussed in this section fall short of the objectives discussed in Section 1.5.2 and motivate the need for a new methodology. Combinatorial optimization methods do not attain the appropriate level of fidelity in characterizing the multiobjective behavior of competing concepts. The sequential optimization methods overcome this limitation but are often infeasible when expensive

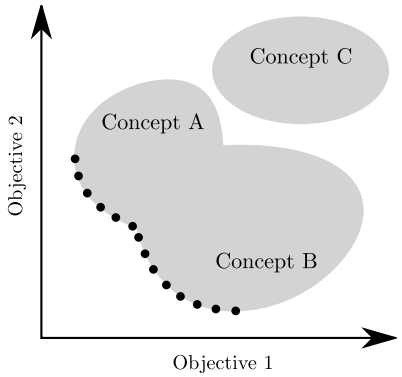


Figure 14: Simultaneous Multiobjective Optimization

codes are used. Considering each design in isolation ignores the relative performance between concepts and wastes resources developing accurate Pareto frontiers regardless of global performance. For many algorithms designed specifically for single-concept optimization, analysis in isolation is the only option as concept may have vastly different sets of design variables.

The most recent developments in simultaneous concept optimization partially overcome this limitation by considering concept performance in the context of competing alternatives. Suboptimal concepts are avoided as designs are located along the s-Pareto frontier. However, this approach is still infeasible for problems of sufficient size and cost. In locating evenly spaced points, these methods view all designs along the frontier as equally informative to the conceptual designer [103]. If a limited budget of function calls leads to a sparse set of optimal designs, a situation may arise where important areas of the space are underrepresented (similar to that depicted in Figure 5a). Rather than reduce uncertainty evenly over the Pareto frontier through simultaneous multiobjective optimization, samples would ideally be located to reduce uncertainty in areas of the design space where selection decisions are most costly. When execution of the analysis code is expensive, the designer is willing to sacrifice accuracy in suboptimal or otherwise less informative areas. A new methodology for multiple competing concept evaluation must be developed to address these needs.

CHAPTER III

ALGORITHMS FOR SOLVING THE MULTIOBJECTIVE PROBLEM

The purpose of this section is to present background into multiobjective problem (MOP) solving. First, a general discussion of multiobjective optimization will be presented followed by some current methods for solving such problems. The chapter will conclude with techniques for optimization of expensive models which will serve as building blocks to a new methodology.

3.1 Pareto Finding Algorithms

The most general form of the constrained multiobjective optimization problem is described by Equations 5-8.

$$\min_x [f_1(x), f_2(x), ..., f_n(x)] \tag{5}$$

$$\text{Subject to:} \quad g(x) \geq 0, \tag{6}$$

$$h(x) = 0, \tag{7}$$

$$x_1 \leq x \leq x_u \tag{8}$$

where f_i is the i^{th} objective, n is the number of objectives, g is a vector of inequality constraints, h is a vector of equality constraints, and x is the vector of design variables.

For many well behaved problems with continuous design variables, there may exist an infinite number of solutions that satisfy Equation 5. The methods discussed in this section are designed to find a representative set of Pareto optimal design points.

3.1.1 Monte Carlo Simulation

Perhaps the simplest Pareto finding algorithm is a technique of random sampling followed by filtering of the results, or Monte Carlo Simulation (MCS). The design variables are randomly sampled from probability distributions to create a single design vector. This vector is then run in a particular analysis to obtain the objective values for that particular design. This process is repeated n times until a representative set of points spanning the entire design space is obtained or computational budget is exhausted. If sufficient computational resources are available or simulation is inexpensive, MCS can produce through filtering, the Pareto frontier for a concept with a high level of accuracy. For expensive simulations, the computational budget does not allow for such a dense representation of the design space. A large number of design variables quickly creates an intractable problem where the number simulations required for adequate resolution can approach tens or hundreds of thousands [127]. At that point, even the task of filtering and sorting the Pareto dominant points becomes computationally infeasible.

3.1.2 Aggregate Objective Function

Another approach to generating Pareto frontiers is sequential optimization of an aggregate objective function. The multiple competing objectives can be functionally combined to form one objective function. The problem is then reduced to iteratively solving a single objective problem using the most preferred search technique. Global optimization methods such as genetic algorithm (GA) [160], simulated annealing [89], particle swarm [49], and others have proved very successful for this type of problem.

Mathematically, the functional form of this aggregation is arbitrary, but a common and perhaps more intuitive approach is the weighted sum of objectives. Weighted sum methods are similar to the weighted decision matrices described by Equation 1. The overall objective to be maximized (or minimized) is a linear combination of multiple

objectives. The coefficients represent the importance of each objective to a decision maker. The general form of the weighted sum of objectives is shown in Equations 9 and 10.

$$S = \sum_{i=n}^{n} \alpha_i f_i(x) \tag{9}$$

$$\text{Subject to} \qquad \sum_{i=1}^{n} \alpha_i = 1 \tag{10}$$

where α_i is the weight on i^{th} objective, f_i is the value of the i^{th} objective, n is the number of objectives, and x is a vector of design variables. These methods require *a priori* specification of preferences before a design can be found. They are often referred to as Utility Functions, Scalarization, and Overall Objective Criterion (OEC) based methods. By randomly assigning values to the α_i's and solving the single objective problem multiple times, unique solutions along the Pareto frontier can be found. The optimization routine must be run n times to produce n Pareto points. Once again, because of the form of the objective function, this approach cannot find solutions in concave regions on the frontier [37, 116]. Because of this, there is no guarantee to get an accurate representation, and optimal solutions may be missed. The work by Athan and Papalambros can also be applied to the aggregate objective function approach to find these deceptive solutions [18].

Another limitation that makes this approach unfavorable for many MOPs is, even with a uniform distribution of weights there is no guarantee to get a uniform distribution along the front [37]. Recent developments have overcome this drawback through introduction of constraints. Kim and de Weck use as adaptive weighted sum method utilizing a series of inequality constraints [87]. Williams et al. developed the ϵ-constraint method by minimizing one objective $y_1(x)$ subject to $y_2(x) \leq \epsilon_q$. Pareto optimal designs can then be generated by sequentially solving the constrained minimization problem for various values of ϵ [185].

The Normal Constraint Method by Messac et al. generates well-distributed Pareto points regardless of the concavity properties of the frontier [101, 105, 107]. Das and Dennis also address the limitation of the aggregation methods in the Normal Boundary Intersection Method (NBI) [38].

3.1.3 Multiobjective Evolutionary Algorithms

Rather than solve a sequence of single objective problems, the goal of MOEAs is to generate a set of Pareto optimal designs concurrently. They are an iterative class of optimization methods where information about the design space is propagated to later 'generations' of designs. The populations evolve to more optimal sets of solutions. One of the earliest introduced evolutionary algorithms for multiobjective optimization was the Vector Evaluated Genetic Algorithm (VEGA). VEGA combines subpopulations from each objective into a single population on which standard crossover and mutation operations can then be performed [150]. This technique turns out to be functionally the same as optimizing a weighted sum of objectives. Because of this, it cannot find solutions on concave portions of the Pareto frontier and biases much of the population towards specific regions of the space. VEGA also suffers from a lack of elitism: non-dominated members are not guaranteed to propagate to later generations which decreases efficiency [42]. There have been many improvements in evolutionary, multiobjective algorithms since the development of VEGA [56, 193], including the Non-dominated Sorting Genetic Algorithm (NSGA) [164]. NSGA was developed to specifically address VEGA's bias towards some solutions on the Pareto frontier, so it provides a more even distribution of points. Additional improvements in efficiency were achieved with the creation of NSGA-II [42], perhaps the most widely used multiobjective optimization algorithm. Improvements in efficiency are gained through a more efficient sorting algorithm, and to many, NSGA-II represents the state of the art of multiobjective evolutionary optimization.

3.1.4 Summary

The methods described above can be used in a sequential optimization and selection framework where each concept is analyzed independently. As mentioned previously, this introduces serious inefficiency as expensive optimization is required for each concept regardless of performance relative to an s-Pareto frontier. As number of objectives and design variables grow, single-concept optimization can represent a significant amount of computational work. One way to address this is to increase the speed of each independent analysis thus reducing the total computational cost across all concepts. This motivates the next class of optimization techniques where the expensive analysis codes are seen as "black boxes" and the designer is most concerned with the inputs (design variables) and outputs (objectives/constraints) rather than inner workings of a particular code.

3.2 Optimization of Expensive Black-Box Functions

While the most widely used multiobjective optimization methods perform well at generating Pareto frontiers, they do so at the expense of many function calls, often numbering in the thousands. When computational resources are at a premium (function call budget of $\sim$100), methods discussed in the previous section may be ill-suited for the task and more advanced optimization schemes are required [92].

3.2.1 Ordinal Optimization and Soft Computing

Similar to qualitative methods for concept evaluation, ordinal optimization (OO) addresses the large amount of uncertainty in early design. Many of the so called real variable-based methods offer concrete analytical results that are very expensive to obtain (first and second order gradient-based or global optimization methods) and may not be necessary. When designers consider problems with larger design spaces, less structure and more uncertainty, the search for a *good enough* design often prevails

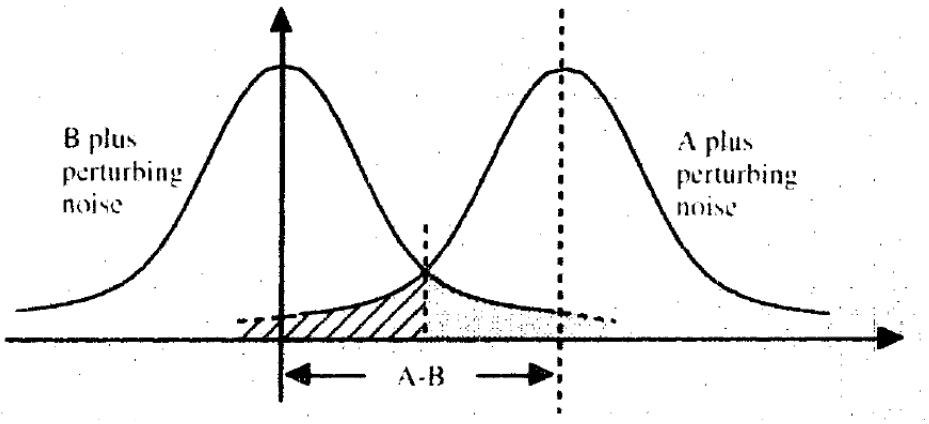


Figure 15: Concept Comparison with Ordinal Optimization [76]

over precise system performance. This is the approach adopted by OO (also called soft computing) which is more concerned with the relative rank order of competing alternatives rather than the accurate values of objectives. OO is based on two tenets [76]:

1. "Order" converges much faster than "value." In other words, it is much easier to determine if A is better than B than to determine $A - B = ?$. This makes ordinal optimization particularly well-suited for expensive function calls. The example shown in Figure 15 illustrates the idea of order versus value. Uncertainty of performance for two concepts A and B is obtained from two simulation experiments. The chance that A and B are misordered is proportional to the area of overlap of these distributions.

2. Goal softening makes finding the optimum easier. When a combinatorial set of potential solutions is considered, traditional optimization is concerned with finding the best with the hopes that it coincides with the true optimum. If the designer is willing to settle for a good enough solution, the probability of achieving successful designs is much greater and requires less computational burden.

For problems with continuous variables, Romero et al. present a Monte Carlo method for probabilistic OO [139]. The goal of the method is to use the minimum number of correlated random samples to deduce ordinality among a set of alternatives. To increase efficiency, the method tries to avoid evaluating samples that serve only to confirm what is already known about ordinal dominance along a single objective.

While ordinal optimization represents an efficient method for establishing dominance among a set of design concepts, it does so at the expense of some valuable information. In the earliest stages of conceptual design, the designer searches for *where* concepts are preferred as well as the aspects of their design that causes them to be so (rather than just *if* they are preferred). Furthermore, a formulation of OO has not yet been developed to address Pareto optimality where rank depends heavily on objective preferences.

3.2.2 Fast Probability Integration

Another method that combines quantitative analysis with uncertainty is Fast Probability Integration (FPI). FPI originated in the 1950's in the field of structural reliability with the goal of minimizing probabilities of failure. More recently, NASA Glenn Research Center (formerly NASA Lewis) incorporated FPI methods into the NESSUS software package which coupled finite element, Monte Carlo, and numerical integration techniques to estimate reliability [68, 112].

The goal of FPI is to efficiently estimate response quantiles as a function of uncertain variables using expensive, differentiable models [69]. This has particular use in early design where degrees of freedom may have large uncertainty distributions. This is illustrated by Figure 16 for a response $f(\mathbf{u})$ as a function of two uncertain design variables u_1 and u_2. When the design variables are characterized by joint probability distributions, FPI can be used to estimate the probability of achieving a certain

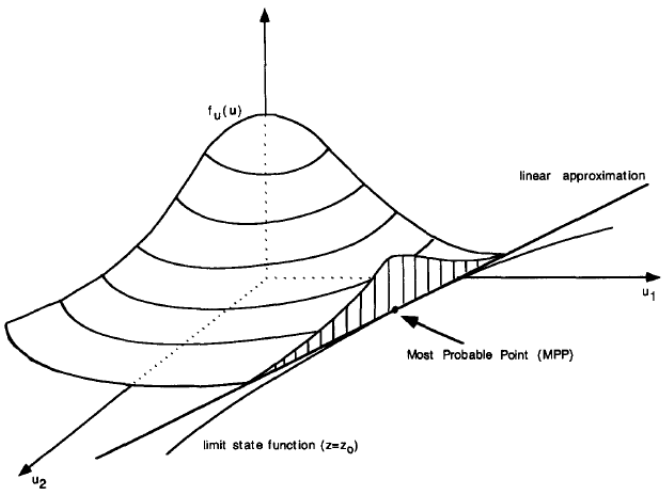


Figure 16: Fast Probability Integration [69]

output value given a random sampling. FPI introduces efficiency gains over multidimensional numerical integration by transforming the design variables into mutually independent, normally distributed, standardized (zero mean, variance of one) random variables. FPI techniques are based on the Most Probable Point (MPP) method [46] where a most probable design is found corresponding to the likeliest output scenario given a limit state function (LSF) z. The LSF represents the desire for a certain output to be above/below some critical value. The most probable point is then the peak of the joint probability distribution that still satisfies the limit function.

FPI can be used in place of Monte Carlo methods and numerical integration for finding a most probable design and obtaining the cumulative distribution function for single objective optimization. Similar to ordinal optimization, the analysis technique used by FPI does not consider tradeoff between multiple objectives. Characterization of concept Pareto frontiers is still required to address this need.

3.2.3 Surrogate-Based Optimization

Surrogate modeling is an approximation method for expensive analyses. The designer can create a surrogate model (or metamodel) and perform simulations with the surrogate much faster than executing the expensive analysis code. Surrogate models

are created by running a representative set of sample points on the expensive code itself. A surrogate can then be 'trained' using the data to give performance predictions close what would be provided by the full model [117]. The types of surrogates range in complexity and predictive capability from simple polynomial expressions (response surfaces) to interpolating functions modeled after the human brain (neural networks). Surrogate models are powerful enablers for many design tasks such as sensitivity analysis and optimization [86, 157, 159, 178].

The most straightforward approach for surrogate-based optimization is simply to replace the expensive code with the surrogate model and use any desired technique for optimization. Wilson et al. execute a dense grid of design variable settings on both polynomial approximations and Kriging models to find Pareto optimal designs [186]. NSGA-II is applied to Kriging by Voutchkov and Keane for structural optimization [174]. These methods are sometimes referred to as two-stage approaches:

1. Create surrogate model

2. Perform optimization

Implicit in these approaches is the fact that validation must be performed with the optimum obtained from the surrogate model. The success of a two-stage approach for optimization is highly dependent on the accuracy of the surrogate, and surrogate accuracy is in turn dependent on the set of training data run in the expensive simulation. Two approaches can address this issue. The first approach is to increase the density of the initial sample which increases accuracy across the entire design space and thus accuracy near the optimum. Unless global accuracy is desired, this can be inefficient as poor performing regions will be sampled just as densely as optimum regions. A dense initial sample is often the best an engineer can do, not because accuracy is desired everywhere, but the locations of optima are unknown *a priori*.

The second approach and one that serves as a key enabler for expensive analysis

codes and this research is *adaptive sampling*. With this technique, surrogate models are sequentially trained as new information about the design space becomes available. Sample N+1 is selected using the locations and responses of all N previous samples. The fundamental assumption with adaptive sampling is that a designer is willing to sacrifice accuracy in suboptimal regions in order to increase predictive capability in desirable areas. For the task of surrogate-based optimization, the most dense sampling of designs would ideally be near the optimum. This ensures that the optimum is true with respect to the expensive analysis code and not an artifact of a poor surrogate prediction. The two-stage method is compared with adaptive sampling for optimization in Figure 17. The red dots are samples run in the expensive analysis, the black line is the surrogate prediction, and the gray area is the confidence in the surrogate prediction.

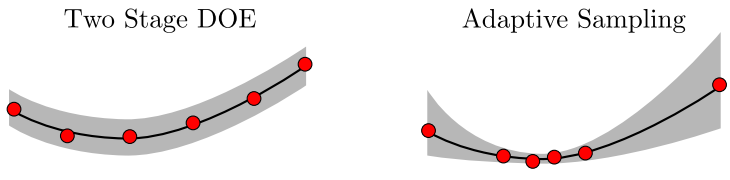


Figure 17: Two Approaches for Surrogate-based Optimization

Adaptive sampling is seeing more widespread use in the engineering community as more expensive analysis codes are used in design and optimization. In the Adaptive Response Surface Method (ARSM) developed by Wang et al., a sequence of response surfaces are used to gradually home in on the optimum. Design variable ranges are sequentially reduced until the region surrounding the optimum is modeled accurately with polynomial equations [177, 179]. Response Surfaced-based Optimization (RSO) is an iterative procedure where a least-squares approximation is developed for an expensive analysis then optimized. The surrogate optimum is then validated in the expensive model, and this new point becomes the next sample point with which to train the surrogate. The process repeats until the surrogate and model optimum

converge [172].

These types of methods have a major disadvantage as they tend to only exploit what is already known about the space and can easily miss the global optimum of a deceptive function. A preferable approach would be to use a technique that is both exploitative and explorative. In other words, it would be desirable to strike a balance between what is known about the current best points while also searching in unexplored regions, where the optimum *may* exist [162].

Trust-region optimization is another type of adaptive sampling that imposes move limits and bounds on a global search. If the optimization moves outside of the appropriate ranges of the surrogate, new samples are evaluated. Dennis and Torczon introduced a trust-region framework for surrogate-based optimization [44]. In this framework, the approximation model and optimization algorithm can be adapted based on improvement or degradation at each step. Zhou et al. search multiple levels of surrogate models to accelerate evolutionary optimization [191]. A global search identifies promising individuals in the population. A gradient-based, trust-region optimization is then executed on those individuals using a local surrogate model. Hughes presents a multiobjective optimization method that searches a binary tree of explicit measures for exploration and exploitation [79]:

- Exploration: Next point is generated within the largest empty region,
- Exploitation: Next point is generated within the largest empty region that is within a small distance of a selected good point.

The methods discussed to this point use either function evaluations or design variable values directly to guide sampling. A third type of adaptive sampling technique makes use of Bayesian models to provide a probabilistic measure of optimality and place designs in areas that are *probably* best.

3.2.4 Bayesian Adaptive Sampling

Bayesian modeling is not a particular type of surrogate model but rather a way of thinking about uncertainty. A Bayesian view, as opposed to frequentist, looks at an uncertain event and represents uncertainty about the event ocurring in terms of probabilities. Consider the example provided by Bishop [25] where we would like to know whether the polar ice caps will disappear by the end of the century. It does not make sense (never mind infeasible) to adopt a fequentist viewpoint and sample a million earths to get a probability distribution of ice melting. In this case, there is only one outcome. The distribution does not arise from randomness or repetition error - a situation where a frequentist approach would be appropriate. Rather, we use the same language of probability to characterize our uncertainty about a particular outcome of an uncertain event [122].

In the context of regression, uncertainty is represented in the surrogate model to reflect that it is an approximation of some true, underlying function. A predictive distribution at a new sample point can be inferred from prior observations of the black-box function [36]. Bayesian surrogate models are already used quite extensively in engineering design and adaptive sampling schemes. A particular type of surrogate model called Kriging (also known as Gaussian Process, stochastic process, DACE model, etc.) originated in the geostatistics community and performs particularly well for deterministic computer experiments [99, 145]. For that reason, Kriging models remain one of the most popular for engineering design and optimization of expensive black-box functions [123, 134]. Technical details of Kriging will be provided in a later chapter, but the discussion continues with background into adaptive sampling methods using Bayesian models.

Perhaps the simplest implementation of Bayesian Adaptive Sampling is to use Bayesian prediction of uncertainty directly in guiding future samples. That is, the candidate point with the maximum uncertainty is chosen as the next sample to run

in the expensive analysis code. In this case, predictor uncertainty is used as a *sampling* or *infill criterion*. Rather than use design variable or objective values directly, maximization along the infill criterion selects the next point/s to "fill in" the set of observed data in desired areas.

Using only model uncertainty as an infill criterion leads to a strictly explorative search strategy and ultimately reduces to space-filling sampling after many iterations. This may lead to slow convergence as information about the current optimum is ignored. Again, a balance of exploration and exploitation is ideal. Cox and John use uncertainty and prediction mean of the Bayesian posterior distribution to compute a lower confidence bound of candidate points [32]. Mockus et al. use an infill criterion based on conditional probabilities [113]. Watson and Barnes developed three separate infill criteria for different classes of problems: 1) locating bounded extremes, 2) locating regional extremes, and 3) and minimizing surprises [182]. A Kriging-based approach using a candidate point's probability, or expectation, of improvement was developed by Jones et al. called Efficient Global Optimization (EGO) in 1998 [84]. EGO uses an infill criterion called Expected Improvement (EI) that is both intuitive and easy to present graphically.

Consider a black-box function of one variable, $f(x)$, to be minimized. Since the function is computationally expensive, it is analyzed at only four design parameter settings, (x_1, x_2, x_3, x_4), producing Figure 18. One of the four samples will have the smallest value representing the current best, f_{best}. Now suppose a Bayesian model is fit through the four sample points. A new predicted point, x^*, is not deterministic but rather the realization of a random process and has a distribution given by the predictor. This is represented in the figure by a normal distribution on x^*. A portion of this distribution falls below the current best sample point, represented by the green shaded region in the figure. The fraction of this distribution is the probability that x^* is an improvement over f_{best}, $P[I(x^*)]$. Instead of a probability, the actual amount of

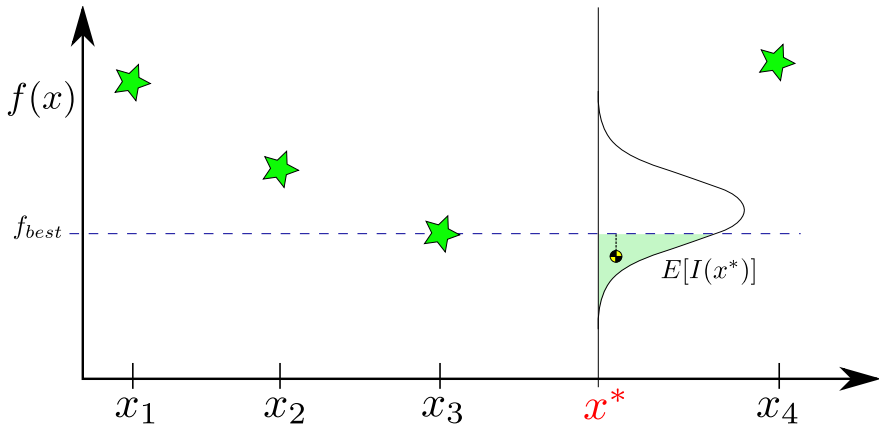


Figure 18: Example Black-Box Simulation Results

improvement that can be expected, $E[I(x^*)]$, can be estimated and shown in Figure 18. $E[I(x^*)]$ can be used as a sampling criterion to be maximized and be evaluated for any point in the design space very rapidly. The next point to sample in the black-box function is the one that maximizes the expected improvement, and this can be obtained from GA or other global optimization method. Expected improvement will be large under two conditions representing exploitation and exploration respectively: 1) the mean of the predicted point is near f_{best} and 2) large standard deviation on the predictor. Both conditions increase the area below the current best sample, so the algorithm is more likely to choose points with these qualities. Two iterations of adaptive sampling using EI are shown in Figure 19 and 20.

The first iteration seen in Figure 19 exploits what is known about the space. The mean of the predicted point that maximizes expected improvement is very close to the current best. In Figure 20, the Bayesian model is updated with this point and expected improvement optimized again. The best point is updated to reflect the information learned from sampling the expensive code at the point identified in Iteration 1. In the second iteration, the algorithm takes a more explorative approach and finds a point where error on the predictor is large. Once the area about the current best point is modeled relatively accurately from the first iteration, uncertainty dominates $E[I(x^*)]$, and the algorithm searches elsewhere.

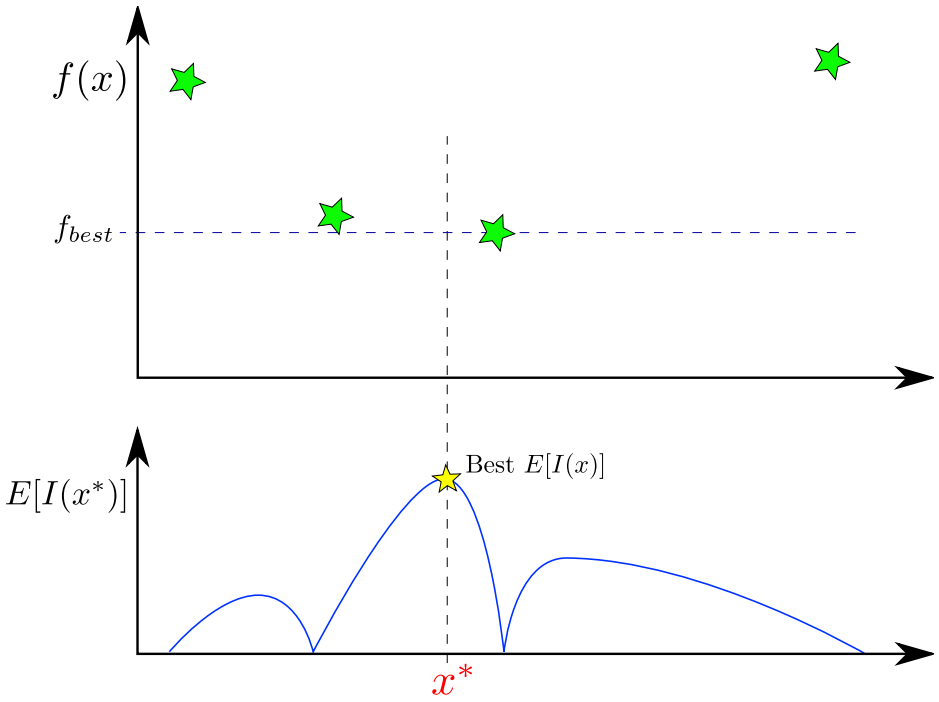


Figure 19: Expected Improvement for Iteration 1

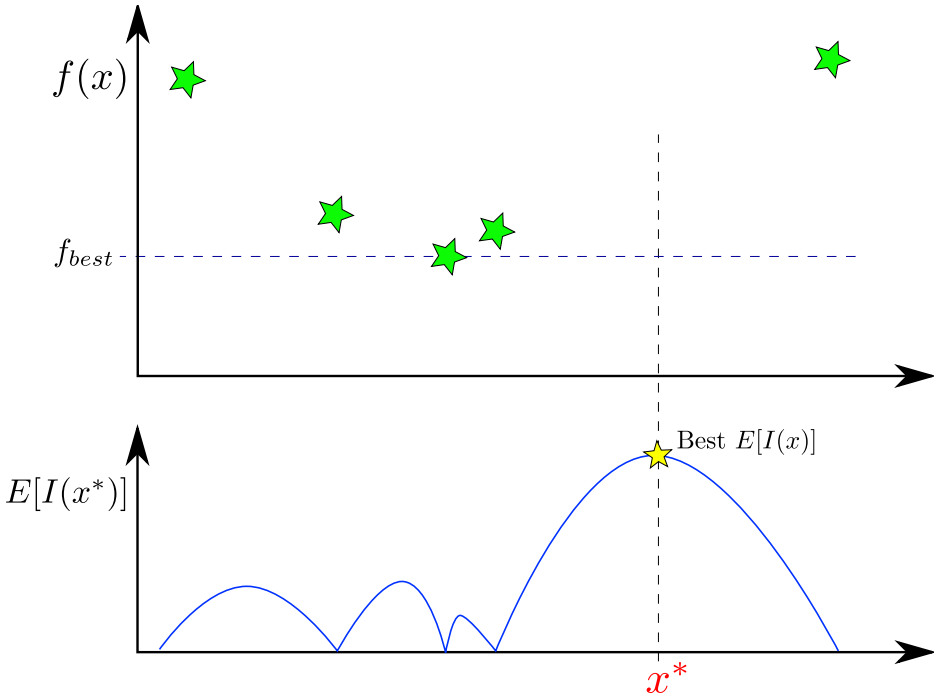


Figure 20: Expected Improvement for Iteration 2

EGO remains one of the most popular Bayesian adaptive sampling techniques, and recent developments in the area have mostly been derivative of this work [60, 72, 83, 173].

3.2.5 Bayesian Adaptive Sampling for Multiple Objectives

Bayesian adaptive sampling has seen some application to multiobjective problems as well. Obayashi proposed *Multi-EGO* that uses an MOEA to find a Pareto optimal set of single objective EIs [121]. Another sequential method is proposed by Knowles called *ParEGO*. The Pareto frontier is gradually built up by converting m separate objectives into a single objective using the augmented Tscebycheff norm [91]. Henkenjohann and Kunert define a new infill criterion based on desirability function which maps the individual objectives to the interval [0,1] [75]. Preferences are then applied to the desirability function to obtain Pareto optimal designs.

Researchers at the University of Southampton present an optimization method for problems with two objectives [85]. Consider the bi-objective example of $f_1(x)$ and $f_2(x)$ shown in Figure 21. A number of analyses have been run, and four of those points are found to be Pareto dominant to the rest of the data set. One more analysis point is run in the expensive function. If this new point falls in the shaded regions, it would augment the current set of Pareto optimal designs. If the point is located in the hatched area, it would dominate and force removal of at least one design from the existing Pareto set.

Now consider that all the sample data is represented by two Bayesian models. One is an approximation of $f_1(x)$, and the other approximates $f_2(x)$. A new predicted point $\mathbf{f}^* = [f_1^*, f_2^*]$ would fall somewhere in the space and can be represented by a posterior distribution on both objectives. The candidate point $\mathbf{f}^*$ is represented by the black dot and the uncertainty given by the Bayesian model as red circles.

A portion of the joint distribution that falls over the hatched area of Figure 22

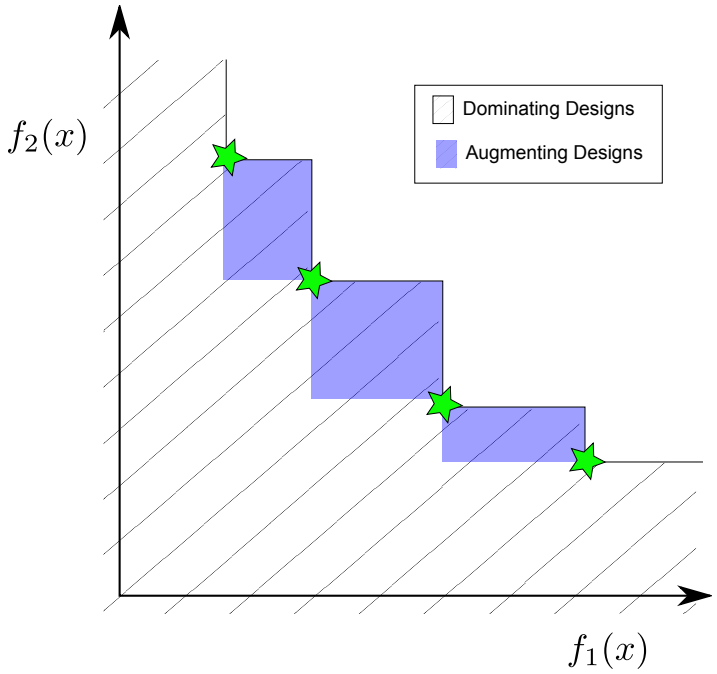


Figure 21: Bi-Objective Optimization of an Expensive Function

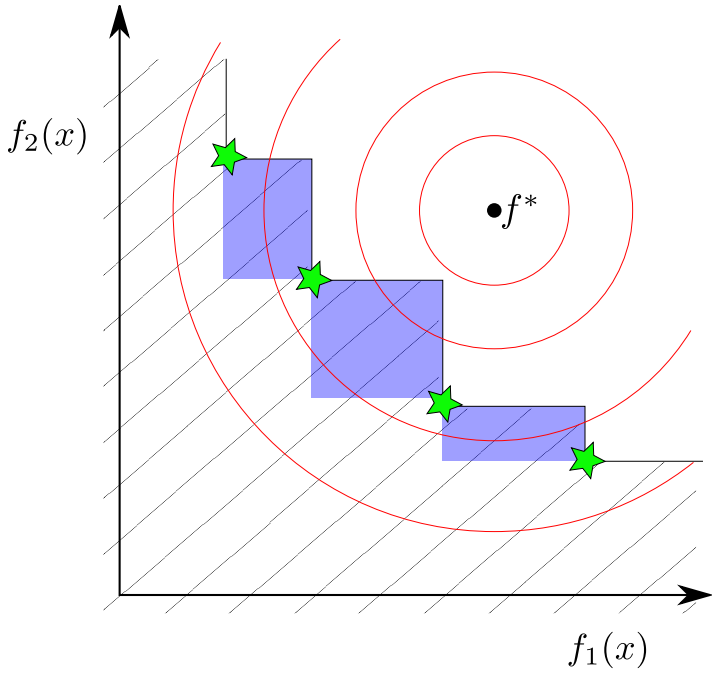


Figure 22: Joint Uncertainty Distribution of a Candidate Design

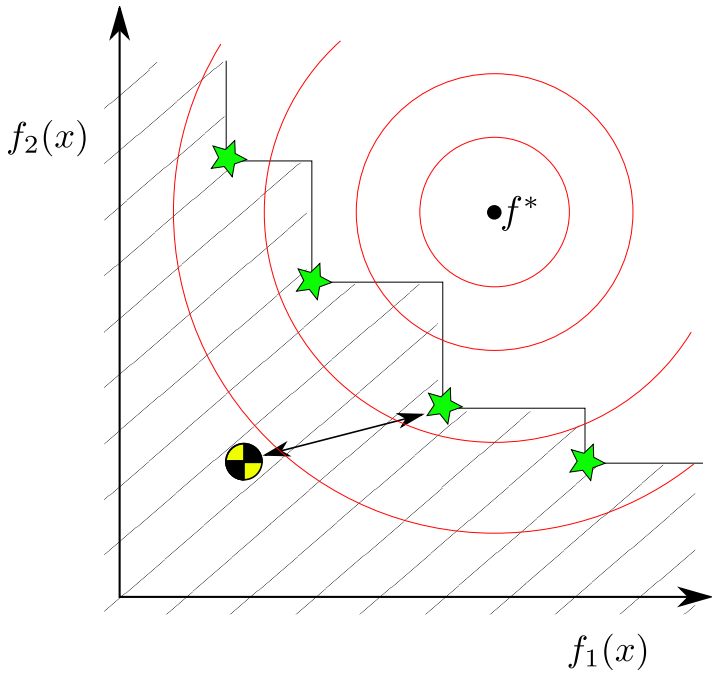


Figure 23: Expected Improvement of Candidate Design

represents the probability that a new point will dominate the current set of designs. The Expected Improvement of that point is then defined as the moment that the first moment of the area (similar to the centroid of the hatched area) makes about the current Pareto frontier. This moment arm and location of the first moment of area are shown in Figure 23. The point that maximizes this moment is the design that is expected to have the largest improvement in both objectives.

In developing the closed-form expression for EI for the bi-objective problem, Keane takes advantage of sorting along the two dimensions. In sorting the Pareto optimal set of designs along the first objective, the second objective is automatically sorted in reverse order. This allows for integration of the hatched area as simply the summation of two-dimensional rectangular areas below and to the right of known Pareto designs (see Figure 24).

Emmerich et al. provide a derivation for expansion into more than two objectives by partitioning the region into a grid [52]. An *enhanced* probability of improvement (PI) method is proposed by Hawe and Sykulski which also uses partitioning of the

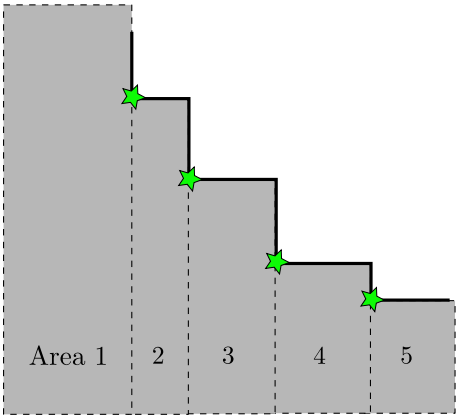


Figure 24: Regions of Integration for the Bi-Objective Problem

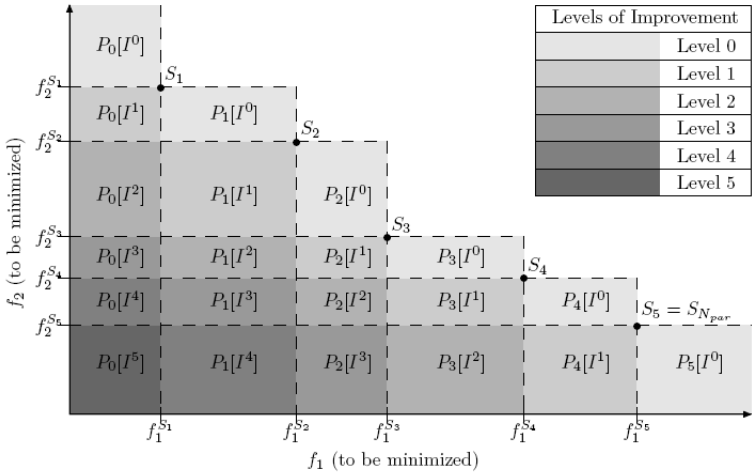


Figure 25: Levels of Improvement for Enhanced Probability of Improvement [71]

space to identify levels of improvement . In [71], k levels of improvement are defined where the k^{th} level of improvement dominates k of the current Pareto optimal designs. These levels of improvement are shown in Figure 25. This method uses PI rather than EI to maintain independence on the scale of objectives.

Other methods of adaptive sampling for multiobjective problems have emerged and often represent incremental improvements over Keane's method [22, 128, 175].

3.2.6 Practical Considerations

While the EGO framework is an elegant solution to global optimization of single and multiobjective problems, practical implementation is not always so straightforward. Real engineering problems introduce new challenges for designers to overcome. For that reason EGO has been extended to handle constraints [29, 82, 136, 149], integer inputs [90], and noisy simulation data [59, 78].

Another important achievement is the capability of handling missing data. This is a common problem when executing simulations of complex systems. A space-filling DOE, no matter how sparse, can unknowingly generate designs with invalid inputs in the model leading to failed cases. Many optimization routines will crash or stall without approaching a global optimum. If an invalid point is chosen in an EGO framework, the algorithm stalls because it has no new sample with which to update the Bayesian model. If the invalid areas are rectangular or otherwise well-behaved, problems can be avoided by just adjusting the ranges or bounds of the independent variables [120]. If the modes of failure are more complex (disconnected or irregularly shaped feasible regions), avoiding such areas with an automated optimization procedure is more difficult [148]. Forrester et al. form a statistical upper bound from the mean $\hat{y}$ and standard deviation $\hat{s}^2$ of the Kriging predictor. A failed point is then replaced by $\hat{y} + \hat{s}^2$ to penalize failed cases and guarantee movement away from regions of infeasibility [57].

3.2.7 Summary

The multiobjective optimization methods presented here can be used effectively as sequential approaches for concept evaluation where individual Pareto frontiers are formed independently. Adaptive sampling and sequential training of surrogate models locate evenly spaced points quite efficiently on or near a Pareto frontier of an individual concept. The methods have also proven quite useful for real engineering

problems supported by expensive models but still suffer from the same inefficiencies as the sequential Pareto finding methods discussed in Section 2.2.2.

CHAPTER IV

THEORY AND FORMULATION

4.1 Research Questions Revisited

This research follows the scientific method for finding a technique for concept evaluation and selection using expensive models. The research questions posed in Section 1.5.3 are revisited followed by a top-level hypothesis to serve as a basis for further research.

Research Question 1 motivated a literature search into the current state of the art of concept evaluation and selection. It was found that a significant gap exists between concept selection and multiobjective decision making with expensive models. Qualitative methods can pare down the combinatorial set of potential concepts to a more reasonable subset of alternatives, but they often do not have the fidelity to adequately characterize the multiobjective nature of the problem. On the other hand, those methods that focus on developing accurate Pareto frontiers are inefficient especially as computational complexity grows for each competing concept. This motivates creation of a new, formal evaluation methodology.

Research Question 2 leads directly to the top-level, methodological hypothesis that describes the proposed solution to the multiobjective decision making problem with expensive models. The hypothesis will lead to lower level research questions and hypotheses as well as supporting computer experiments designed to answer each question.

4.1.1 Pareto Frontier Intersection-Based Concept Evaluation

A new paradigm in concept evaluation based on Pareto frontier intersections (PFI) is introduced which represents the most significant contribution of this research. Up

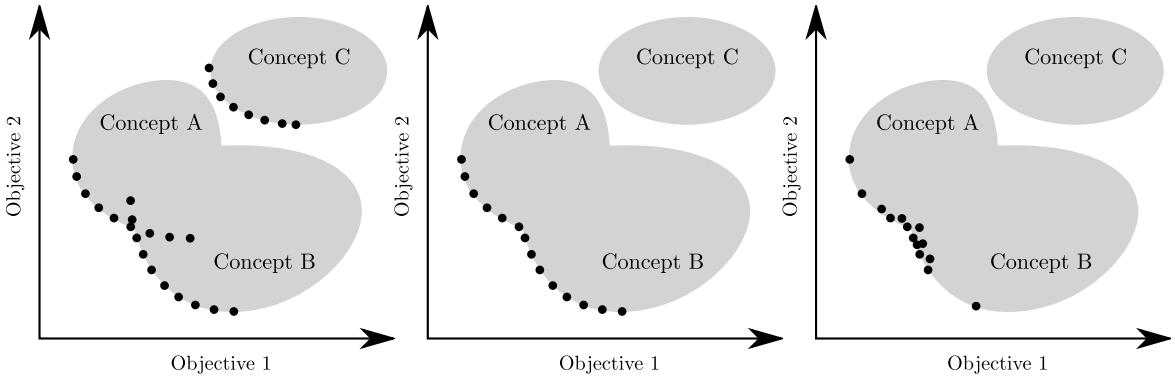
to this point, the methods discussed in the literature focus on either 1) accurate individual Pareto frontiers (sequential optimization) or 2) accurate s-Pareto frontier (simultaneous). These methods share the drawback that designs are evaluated in areas that inevitably dominate all concepts. This is shown in Figures 26a and 26b where the s-Pareto frontier is accurately modeled even towards the edges (best-in-class designs) of the frontier. This paradigm has already proved to potentially yield poor results for concept selection (Figure 5a). As mentioned in Section 1.5.2, the designer would rather focus on those areas where concept selection is less obvious and thus minimize the risk associated with choosing a concept.

This is the focus on PFI-based concept evaluation. The intersection of Pareto frontiers is the location where optimality shifts from one concept to another. PFI-based evaluation focuses first on Pareto frontier intersections thus using a minimum number of function calls to establish *relative* relationships and identify where optimality shifts between concepts. Characterization of the Pareto intersection is shown graphically in two dimensions in Figure 26c.

In order to develop a PFI-based evaluation method, we must first decompose what it means for a design to be located at the intersection of Pareto frontiers. A point that lies on the intersection has two properties: 1) lies on the s-Pareto frontier of all concepts and 2) has similar performance to designs of other concepts. At the precise intersection, there exist multiple concepts that can satisfy the problem for fixed preferences on the objectives.

Hypothesis: A PFI-based concept evaluation methodology will enable multiobjective decision making with independent, expensive models. PFI-based evaluation will be enabled by a balanced search for designs with the following properties:

- Pareto optimal relative to all concepts
- Similar performance to designs of competing concepts



(a) Sequential Optimization (b) Simultaneous Optimization (c) PFI-Based Evaluation

Figure 26: Three Approaches for Concept Evaluation

4.2 Technical Preliminaries

Methodological development begins with some technical details of past work that will serve as building blocks for Pareto Frontier Intersection-based Concept Evaluation and Selection. Introduced in Chapter 3.2, the following techniques will be discussed in greater detail:

- Kriging
- Probability of Improvement

4.2.1 Kriging

Simple Kriging is discussed here with development and notation adopted from the seminal paper by Jones et al. [84]. If we suppose a deterministic function, $y(x)$, of k variables is sampled n times, a simple linear regression model for a new candidate point x can be written in the form shown in Equation 11.

$$\hat{y}(x) = \sum_{h} \beta_h f_h(x) + \epsilon \tag{11}$$

The $f_h(x)$'s are linear or nonlinear functions of the independent variable x, the β_h's are unknown coefficients to be estimated, and the ϵ is normally distributed error. In standard regression techniques such as polynomial approximations, these errors are

assumed independent. This is where Kriging diverges from other modeling techniques. If the output response is from a deterministic computer model, the error is due to modeling error only, not noise or measurement error. It follows then that if the response is continuous, the error is continuous. If two points $x^{(i)}$ and $x^{(j)}$ are close together, so too should their prediction errors. Rather than assume zero correlation between errors at two points, it makes more sense to assume the correlation is related to the distance between the two points. The stochastic process model, as Kriging is it often called, avoids the assumption of independent errors by treating correlation of errors in this way. Rather than Euclidean distance which would weight all independent variables equally, a special weighted distance formula is used shown in Equation 12.

$$d(x^{(i)}, x^{(j)}) = \sum_{h=1}^{k} \Theta_h |x_h^{(i)} - x_h^{(j)}|^{p_h}, \quad (\Theta_h \leq 0, p_h \in [1, 2]) \tag{12}$$

The Θ_h's in Equation 12 represents the importance that particular variable, h, has on the response. The p_h's are a measure of the smoothness in the h direction. Many implementations of Kriging default the p_h's to two assuming smooth behavior in every direction. This leaves only the Θ_h's to tune that constitutes training a Kriging model. Correlation of errors is then given by Equation 13.

$$R[x^{(i)}, x^{(j)}] = \exp[-d(x^{(i)}, x^{(j)})] \tag{13}$$

Intuitively, points close together should be expected to have similar errors. This follows from the forms of the equations where points with small distances between them will have high correlation. As the distances increase, correlation goes to zero. The hyperparameters Θ_h are found by maximizing the likelihood function, lf, given by Equation 14.

$$lf = \frac{1}{(2\pi)^{\frac{n}{2}}(\sigma^2)^{\frac{n}{2}}|R|^{\frac{1}{2}}} \exp\left[-\frac{(\mathbf{y} - \mathbf{1}\mu)' R^{-1} (\mathbf{y} - \mathbf{1}\mu)}{2\sigma^2}\right] \tag{14}$$

where $\mathbf{y}$ is a vector of observed responses, $\mathbf{1}$ is a vector consisting of all ones, and the (i, j) component of the correlation matrix R is given by Equation 13. The variance

σ^2 and mean of the model μ are given by Equations 15 and 16 respectively.

$$\sigma^2 = [(\mathbf{y} - \mathbf{1}\mu)' R^{-1} (\mathbf{y} - \mathbf{1}\mu)]/N \tag{15}$$

$$\mu = \frac{\mathbf{1}' R^{-1} \mathbf{y}}{\mathbf{1}' R^{-1} \mathbf{1}} \tag{16}$$

The Kriging prediction $\hat{y}(x)$ and error on that prediction $\hat{s}^2(x)$ are then given by Equations 17-19.

$$\hat{y}(x) = \mu + \mathbf{r}' R^{-1} (\mathbf{y} - \mathbf{1}\mu) \tag{17}$$

$$\mathbf{r}(x) = R(x, \mathbf{x}) \tag{18}$$

$$\hat{s}^2(x) = \sigma^2 \left[1 + \mathbf{r}' R^{-1} \mathbf{r} + \frac{(1 - \mathbf{1}' R - \mathbf{1} r)^2}{\mathbf{1}' R^{-1} \mathbf{1}} \right] \tag{19}$$

where $\mathbf{r}$ is a vector of correlations between a new candidate point x and the observed data $\mathbf{x}$. The Kriging model can also be written in the form given by Equation 20.

$$\hat{y}(x) = \mu + \epsilon(x) \tag{20}$$

Notice the difference between Equation 11 and 20. The linear regression terms have been replaced by a constant μ, which is the mean of the model. The error terms are no longer independent but normally distributed random variables correlated according to Equation 13. Kriging is often called a stochastic process model because the errors are a stochastic process. This reflects the Bayesian interpretation that the error is not due to randomness, but rather, expressed as a randomly distributed variable because we do not know the exact location of the prediction. Kriging is said to be an interpolating function in that the prediction at a trained point is exact and error goes to zero at that point (within machine precision). Error at an unknown point then depends on distance from known sample points and the true value of the function at those points.

4.2.2 Probability of Improvement

Once a Kriging model is built from the observed data, the Probability of Improvement can be calculated for any new candidate design. Derivation of PI first begins with single-objective formulation then develops equations for two objectives. The section concludes with discussion of how the approach can be expanded to arbitrary number of objectives.

If y_{min} is the minimum response observed so far in an expensive model, we can define improvement at any new candidate point x^* as

$$I(x^*) \equiv \begin{cases} y_{\min} - \hat{y}(x^*) & \text{if } \hat{y} < y_{\min} \\ 0 & \text{otherwise} \end{cases} \tag{21}$$

If the prediction at x^* is not deterministic but represented by a normal distribution like that given by Kriging, the probability of improving $P[I(x^*)]$ is defined as the area of the posterior probability distribution below the current best point. This is given by Equation 22.

$$P[I(x^*)] = \int_{-\infty}^{y_{min}} \hat{\phi}(y)dy \tag{22}$$

where the predicted Gaussian probability distribution function $\hat{\phi}$ (PDF) is defined by

$$\hat{\phi}(y) = \frac{1}{\sqrt{2\pi\hat{s}^2}} \exp\left\{-\frac{(y-\hat{y})^2}{2\hat{s}^2}\right\} \tag{23}$$

where $\hat{y}$ and $\hat{s}^2$ are functions of x^* and given by Equations 17 and 19. Equation 22 can also be written in terms of the normalized Gaussian cumulative distribution function Φ (CDF):

$$P[I(x^*)] = \Phi\left(\frac{y_{min} - \hat{y}(x^*)}{\hat{s}(x^*)}\right) \tag{24}$$

4.2.2.1 Multiobjective Probability of Improvement

The discussion continues with the bi-objective extension developed by Keane at the University of Southampton [85]. A two-dimensional Gaussian PDF can be built from

the posterior Gaussian distribution for the predicted response at x^*.

$$\hat{\phi}(y_1, y_2) = \frac{1}{\sqrt{2\pi\hat{s}_1^2}}\exp\left\{-\frac{1}{2}\frac{[y_1 - \hat{y}_1]^2}{\hat{s}_1^2}\right\} \times \frac{1}{\sqrt{2\pi\hat{s}_2^2}}\exp\left\{-\frac{1}{2}\frac{[y_2 - \hat{y}_2]^2}{\hat{s}_2^2}\right\} \tag{25}$$

In relation to a single observed Pareto point $\bar{y} = [\bar{y}_1, \bar{y}_2]$ the probability that a new point x^* is an improvement in either direction is obtained by integrating the predicted joint PDF over the area below and to the left of that Pareto point.

$$P[\bar{y}_1 \leq y_1(x^*) \cup \bar{y}_2 \leq y_2(x^*)] = \Phi\left[\frac{\bar{y}_1 - \hat{y}_1(x^*)}{\hat{s}_1(x^*)}\right] + \Phi\left[\frac{\bar{y}_2 - \hat{y}_2(x^*)}{\hat{s}_2(x^*)}\right] - \Phi\left[\frac{\bar{y}_1 - \hat{y}_1(x^*)}{\hat{s}_1(x^*)}\right]\Phi\left[\frac{\bar{y}_2 - \hat{y}_2(x^*)}{\hat{s}_2(x^*)}\right] \tag{26}$$

This is just relative to a single Pareto point. In order to get the probability a new point $\hat{y}(x^*)$ dominates at least one member of the *entire* Pareto set, the joint PDF must be integrated over the area below and to the left of this set. If the set of observed Pareto optimal points $\bar{\mathbf{y}}$ is described by Equation 27, the integration over the area of improvement can then be written in the formed shown by Equation 28.

$$\bar{\mathbf{y}} = [(\bar{y}_1^1, \bar{y}_2^1), (\bar{y}_1^2, \bar{y}_2^2), (\bar{y}_1^3, , \bar{y}_2^3), ...(\bar{y}_1^M, \bar{y}_2^M)] \tag{27}$$

$$P[\bar{y}_1 \leq y_1(x^*) \cup \bar{y}_2 \leq y_2(x^*)] = \int_{-\infty}^{\bar{y}_1^1}\int_{-\infty}^{\infty}\hat{\phi}(y_1, y_2)dy_2dy_1 + \sum_{i=1}^{M-1}\int_{\bar{y}_1^i}^{\bar{y}_1^{(i+1)}}\int_{-\infty}^{\bar{y}_2^{(i+1)}}\hat{\phi}(y_1, y_2)dy_2dy_1 + \int_{M}^{\infty}\int_{-\infty}^{\bar{y}_2^M}\hat{\phi}(y_1, y_2)dy_2dy_1 \tag{28}$$

or

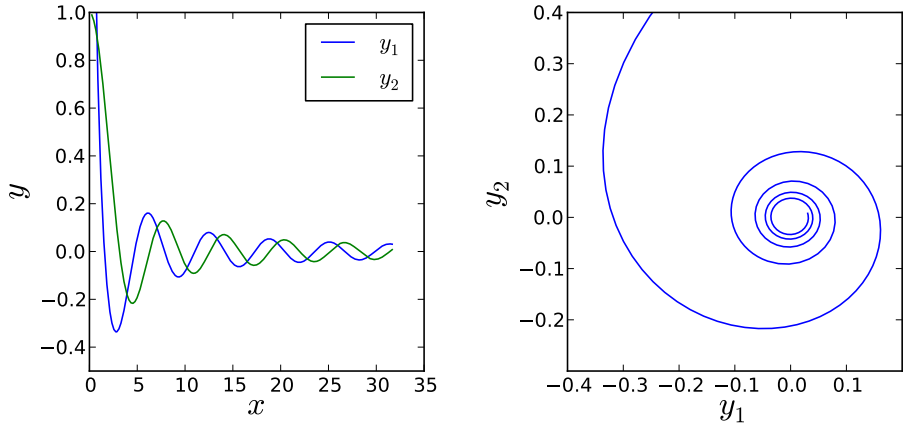
$$P[I(x^*)] = \Phi\left[\frac{\bar{y}_1^1 - \hat{y}_1(x^*)}{\hat{s}_1(x^*)}\right] + \sum_{i=1}^{M}\left\{\Phi\left[\frac{\bar{y}_1^{i+1} - \hat{y}_1(x^*)}{\hat{s}_1(x^*)}\right] - \Phi\left[\frac{\bar{y}_1^{i} - \hat{y}_1(x^*)}{\hat{s}_1(x^*)}\right]\right\} \times \Phi\left[\frac{\bar{y}_2^{i+1} - \hat{y}_2(x^*)}{\hat{s}_2(x^*)}\right] + \left\{1 - \Phi\left[\frac{\bar{y}_1^{M} - \hat{y}_1(x^*)}{\hat{s}_1(x^*)}\right]\right\} \times \Phi\left[\frac{\bar{y}_2^{M} - \hat{y}_2(x^*)}{\hat{s}_2(x^*)}\right] \tag{29}$$

This is a nondimensional number representing the probability that a new point will be Pareto optimal given the set of observed data. This infill criterion will be referred to throughout this thesis as Multiobjective Probability of Improvement (MOPI). The equations presented in this section are closed form solutions for the probability of improvement for bi-objective problems. The summation of integrals in Equation 28 takes advantage of the ability to sort the list of Pareto optimal points. To address problem with arbitrary number of objectives, other solution methods must be employed.

Monte Carlo Simulation is an effective and fast way to find an approximation for MOPI. Random samples can be generated according to the posterior normal distributions provided by the Kriging predictor for each objective. The resulting samples that are dominated by the currently observed set of Pareto points can then be filtered out. The fraction of resulting points to total Monte Carlo cases is an approximation for the candidate point's probability of improvement. As the number of samples becomes large, the approximation approaches exactly what would be obtained from integration. In the author's experience, much fewer samples (on the order of 100's) are required to yield good results. Monte Carlo has the advantage in that the number of sample points can be tuned according to the demands of the problem leading to faster computation. On the other hand, numerical integration can become quite computationally intensive for problems with many objectives.

4.2.2.2 MOPI Example

The performance of an adaptive sampling scheme based on MOPI is shown in a simple example. Consider a model $\mathbf{y}$ of a single independent variable x that produces two outputs y_1 and y_2 that a designer would like to minimize. The model is described by Equation 30.



(a) x vs. y (b) y_1 vs. y_2

Figure 27: Spiral Function

$$\begin{aligned} y_1(x) &= \cos(x)/x \\ y_2(x) &= \sin(x)/x \end{aligned} \quad , \quad 0.25 \leq x \leq 10\pi \tag{30}$$

The plot of y_1 and y_2 as a function of x is shown in Figure 27a. The objective space is shown in Figure 27b with the Pareto frontier as the lower left-most portion of the spiral. Upon inspection, the optimal designs lie in a region of approximately 3 $\leq x \leq$6, where both y_1 and y_2 are minimum.

Suppose the model is sampled eight times and a Bayesian surrogate model fit through each of the two responses. This produces the plot shown in Figure 28. The dots represent initial sample points. The predictions are exact within machine precision at the observed locations, but the Bayesian model does a poor job of predicting the true model in unknown regions. This is even more evident when looking at the objective space in Figure 29. The designer would now like to apply MOPI to locate the next sample point and increase accuracy in Pareto optimal regions.

After ten iterations of adaptive sampling based on MOPI and retraining the Bayesian predictor, Figures 30 and 31 can be generated. In Figure 30, the original sample set is shown along with the adaptively sampled locations. It can be seen

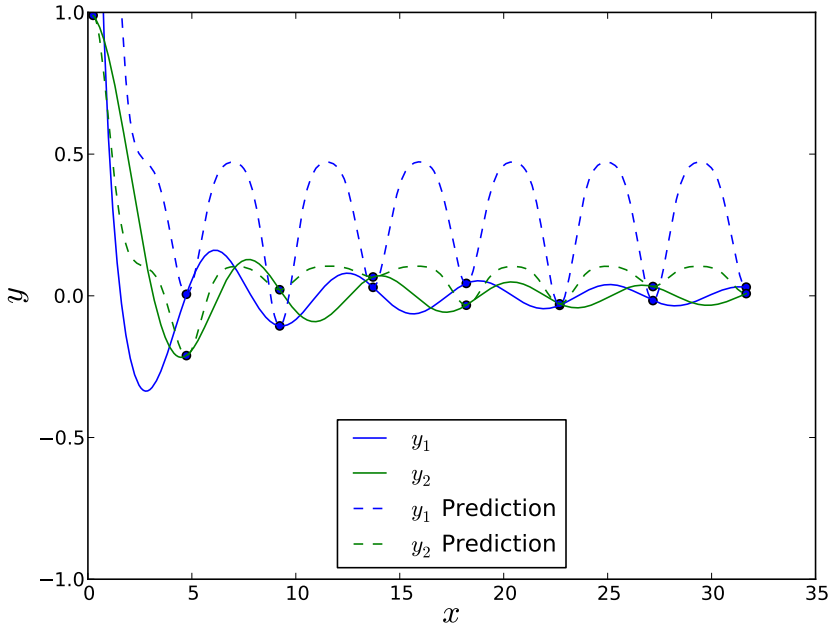


Figure 28: Spiral Function with Surrogate Model Prediction

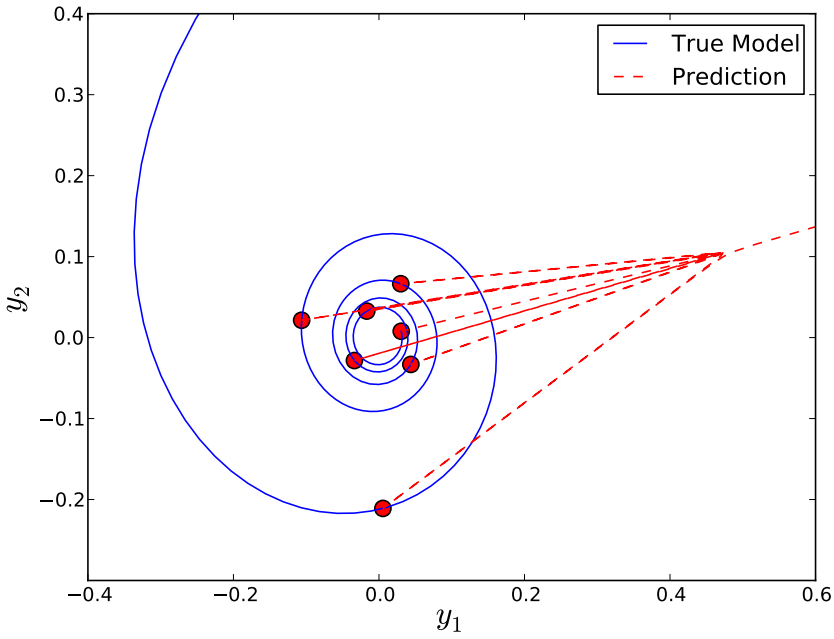


Figure 29: Spiral Function Pareto Space with Surrogate Model Prediction

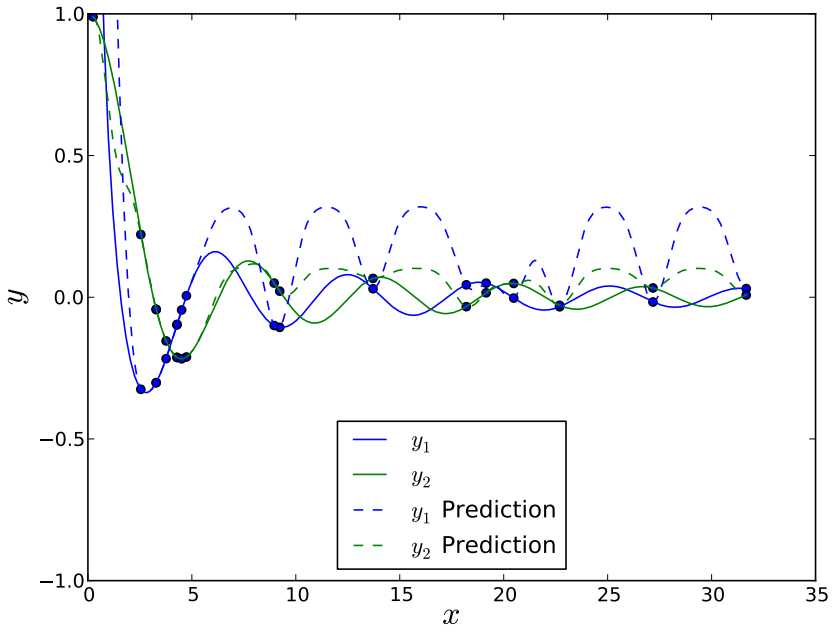


Figure 30: Spiral Function with Surrogate Model Prediction after MOPI Sampling

that many point were placed in the region where $x \approx 4$, the true region of the Pareto frontier where both y_1 and y_2 are small. Notice also that a few points were placed where $x \approx 20$. This highlights the fact it was believed that both functions were small in this region. The Kriging prediction after the iteration is very close to the true function in the region of Pareto optimality and still poor in suboptimal areas. Fidelity is sacrificed here because they *probably* do not contain dominant designs. The Pareto space shown in Figure 31 depicts a very accurately modeled Pareto frontier because samples have been carefully placed to focus on this region.

4.3 Enablers for a New Methodology

The machinery is in place for development of a sampling criterion that will enable a Pareto Frontier Intersection-based evaluation methodology. A new infill criterion called Pareto Intersection Closeness (PIC) will balance two aspects for adaptively sampling the designs spaces of multiple concepts: 1) search for optimal designs across

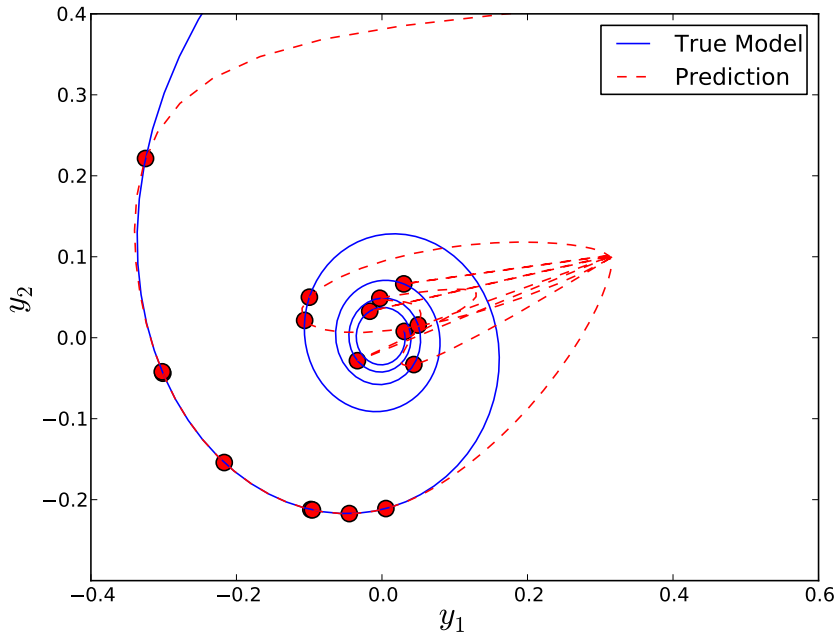


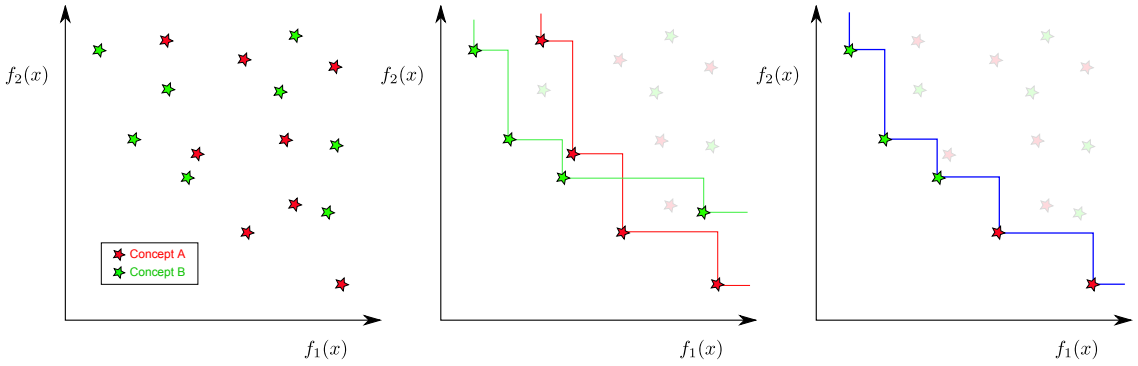
Figure 31: Spiral Function Pareto Space with Surrogate Model Prediction after MOPI Sampling

all concepts (s-Pareto) and 2) similar performance between designs of competing concepts. The criterion that balances these two goals is hypothesized to guide samples into regions where concept selection decisions are most costly.

MOPI combined with Kriging surrogate models presented in the previous section represent a *sequential* evaluation approach where each concept must be evaluated in isolation. Development of PIC begins with modification to classical MOPI called Multi-Pareto Probability of Improvement (MPPI) which addresses the goal of finding designs that are on the s-Pareto frontier. The second objective (locating similar performing designs) will be addressed with a measure of closeness called Normalized Pareto Distance (NPD).

4.3.1 Multi-Pareto Probability of Improvement

Classical MOPI is now modified to account for multiple competing concepts and drive designs into regions of s-Pareto optimality. The distinction between optimal and



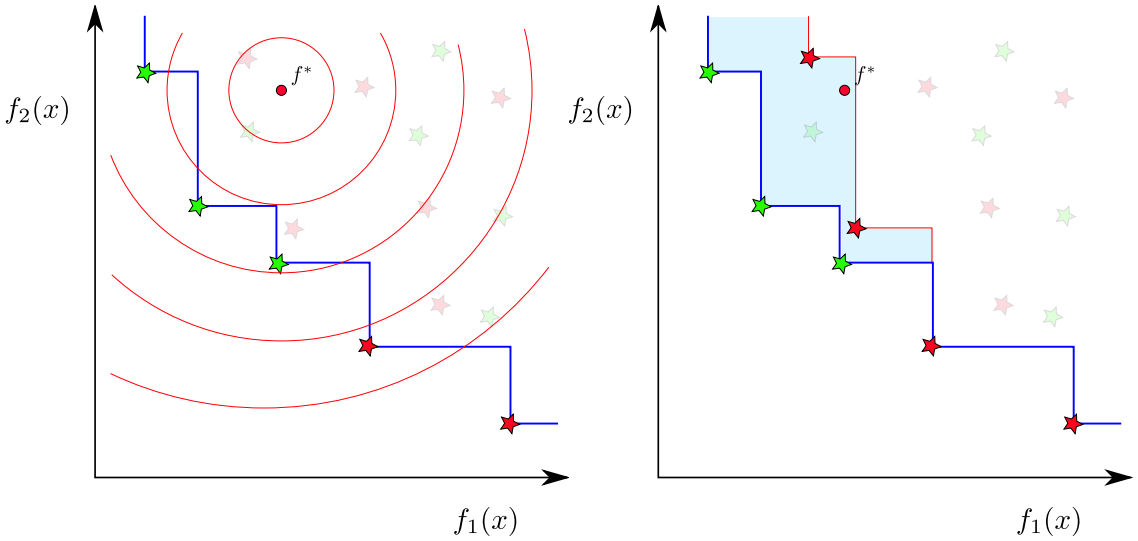
(a) Multi-Concept Sampling (b) Individual Pareto Frontiers (c) s-Pareto Frontier

Figure 32: Multiple Concept Sampling

suboptimal is now made. Designs are considered optimal if they dominate designs from all other concepts and fall on the s-Pareto frontier. Designs on Pareto frontiers of competing concepts but *not* on the s-Pareto frontier are considered dominated or suboptimal.

MPPI begins with a graphical depiction shown in Figure 32. Consider the minimization problem for two concepts and two objectives f_1 and f_2 as a function of respective design variables. Observations for each concept are available and shown in Figure 32a by green and red stars. The Pareto frontiers of each concept can be drawn like those depicted in Figure 32b. There are portions of each frontier however, that are dominated by the other concept forming an s-Pareto frontier (blue line in Figure 32c).

Surrogate models can be fit through each objective for each concept for a total of four approximation models. Now suppose a new point f^* is predicted at a random setting of the design variables for Concept A as in Figure 33a. If the prediction is a randomly distributed variable, such as that given by a Bayesian model, a portion of the uncertainty distribution for that point will lie below and to the left of the current s-Pareto frontier. Using an infill criterion based on the dominant Pareto points has the following consequence. In regions that are Pareto optimal for a single concept but not multiple concepts, there is a reduction in area under which the joint probability



(a) New Predicted Point (b) Dominated Region

Figure 33: Development of a New Sampling Criterion for Multiple Concepts

distribution is integrated. The amount of reduction is shown by the shaded region of Figure 33b. Points in this region will have a lower MPPI compared with MOPI because of the contribution to the Pareto dominant set of designs from Concept B. If the new prediction is close to an already run dominated design, MPPI will be reduced. This is a result of the fact that objectives are approximated by surrogates that assume correlated errors. Points close to already run designs will have small errors leading to a small volume of the joint PDF that lies in the Pareto dominant region. Mathematically, the calculation of MPPI is done using the same Equation 28. The Pareto set of points $\bar{y}$ may now be made up of designs from multiple competing concepts.

4.3.2 Normalized Pareto Distance

In addition to Pareto optimality, a mechanism is needed to drive designs of competing concepts to be similar in performance along the multiple objectives. A measure of current Pareto frontier closeness will be used called Normalized Pareto Distance (NPD). NPD is a measure of distance between a candidate point $\hat{\mathbf{y}} = [\hat{y}_1, \hat{y}_2, ..., \hat{y}_n]$

at x^* and a reduced s-Pareto set of points formed *without* the current concept under evaluation. NPD for n objectives is given by Equation 31.

$$\text{NPD} = \min \left[\sum_{i=1}^{n} \frac{\bar{y}_i^j - \hat{y}_i(x^*)}{\bar{y}_i^j} \right], \quad j = (1, 2, ..., M) \tag{31}$$

where $\bar{y}_i^j$ is the j^{th} Pareto optimal point along the i^{th} objective.

NPD is essentially the sum of fractional errors of each objective between a candidate point and the closest optimal design from a competing concept. Minimizing NPD will find the design that is closest to other concepts' current optimal points across all objectives. This formulation is used rather than Euclidean distance to ensure independence to the scales of the objectives.

Note that nothing in the formulation of NPD guarantees Pareto optimality. The designs that form $\bar{y}$ are only Pareto optimal points given the previously run data. These points may or may not lie near the true Pareto frontier. For this reason, NPD is a purely exploitative criterion that focuses samples around *current* designs without searching for those that are potentially more optimal.

4.3.3 Pareto Intersection Closeness

An aggregate infill criterion combining MPPI and NPD balances search for Pareto optimality with similar performance between designs. The new infill criterion, called Pareto Intersection Closeness (PIC) is defined in Equation 32.

$$\text{PIC} = \text{MPPI} - \text{NPD} \tag{32}$$

where subtraction appears in the definition to be consistent with maximization of the infill criterion.

4.3.4 Further Considerations

4.3.4.1 Constraints

In assuming the problem at hand is optimization of expensive codes, it can also be assumed that constraints are also expensive to evaluate. This methodology for

PFI-based evaluation will take the approach presented by Forrester and Keane in handling constraints probabilistically [60] where both the objectives and constraints are assumed modelable. If a candidate point modeled with a Kriging predictor resides with low uncertainty in an infeasible area, the probability of improvement should be expected to be low to reflect a constraint violation.

Rather than calculating the probability of improving relative to the current best, the probability of the prediction being greater than the constraint limit can be calculated in much the same way using Equation 28. This can be written for a general inequality constraint $[g(x) > 0]$ as

$$P[g(x) > 0] = \int_0^\infty \hat{\phi}(G)dG \tag{33}$$

where G is a random variable, g is the constraint function, and $\hat{\phi}(G)$ is given by

$$\hat{\phi}(G) = \frac{1}{\sqrt{2\pi\hat{s}^2}}\exp\left\{-\frac{(G-\hat{g})^2}{2\hat{s}^2}\right\} \tag{34}$$

where $\hat{g}$ and $\hat{s}^2$ are given by the Kriging prediction on g.

The probability of improving *and* being feasible, assuming independence, is then given Equation 35. This is also extensible to many constraints.

$$P[I(x) \cap g(x) > 0] = P[I(x)] \times P[g(x) > 0] \tag{35}$$

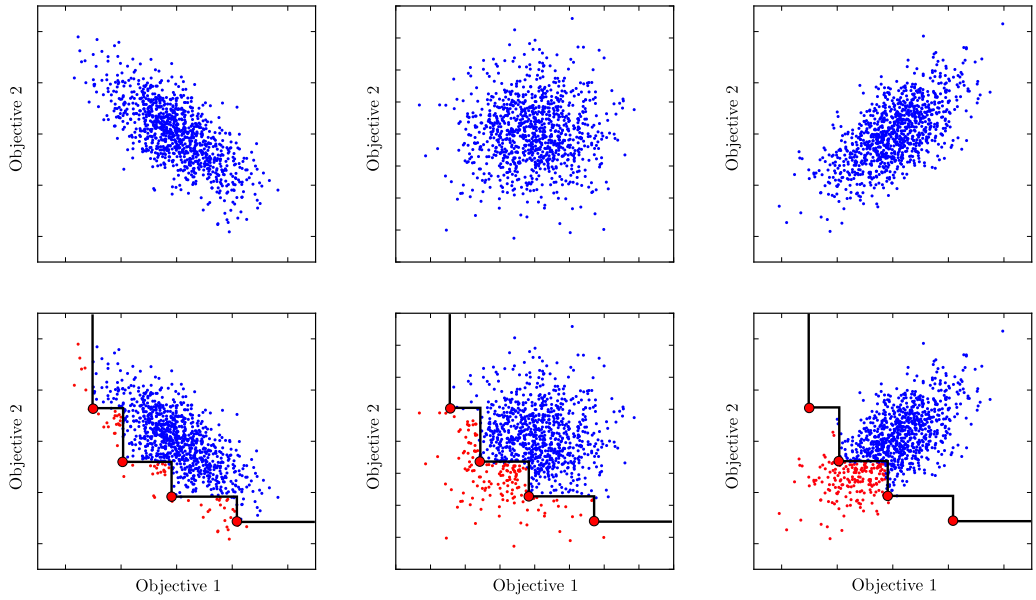
The adjustment to probability of improvement at a new design point x^* can be incorporated into PIC in the manner described by Equation 37 where the final formula takes into account an arbitrary number of objectives n.

$$PIC = P[I(x^*)] \times P[g_1(x^*) > 0] \times P[g_2(x^*) > 0] \times ... \times P[g_n(x^*) > 0] - NPD(x^*) \tag{36}$$

or

$$PIC = MPPI \times PF - NPD \tag{37}$$

where PF is the probability of being feasible (satisfying *all* constraints) and defined as the multiplication of all the individual probabilities for each constraints.



(a) Negative Correlation (b) Zero Correlation (c) Postive Correlation

Figure 34: Correlated Objectives for MPPI Estimation

4.3.4.2 Objective Correlation

In many examples of modeling and simulation of complex systems, the outputs of a computer code are not independent but in fact correlated in some way. If this information is available, either through prior experience or available simulation data, it can be used to further enhance the estimate for probability of improvement and guide sampling into regions that are truly optimal across all concepts. For incorporation into the PFI-based evaluation methodology, the MPPI can be approximated by generating n random, correlated samples of a normally distributed variable given by the Kriging predictor. A candidate design's *probability of improvement* can then be calculated as the percentage of random samples that improve upon the current set of best designs. This is illustrated in Figure 34.

Three cases are presented in the figure: negative, zero, and positive correlation. A candidate point given by Kriging is the realization of a random process and is

assumed Gaussian with mean and standard deviation given by Equations 17 and 19 respectively. The assumed independent objectives can be augmented with correlation information and alters the estimation for MPPI as shown in the bottom row of Figure 34. The large red dots are known s-Pareto optimal solutions from prior evaluations of the concept models. The black line represents the Pareto boundary. Any evaluated design that falls below and to the left of the black line (small red dots) will augment the current set of optimal designs.

4.3.4.3 Implementation

In computing PIC, elements of the EGO algorithm (Jones et al. [84]) are used, and many of the calculations are presented in a Matlab code by Forrester under the GNU Lesser General Public License (LGPL) [58]. The Forrester code was re-implemented by the author in Python computer language inside of an open-source computational framework called OpenMDAO. This framework, developed by NASA and currently in an alpha release, is specifically designed to efficiently integrate analysis tools and methods [10].

Another requirement for the method is an implementation of a Kriging surrogate model generator within OpenMDAO. While the limitations of Kriging are well documented [51, 99], it is assumed this type of Bayesian model will be appropriate for the proposed method. To alleviate some of the ill-conditioning issues commonly encountered in forming Kriging metamodels, Cholesky decomposition is used to invert the correlation matrix formed by Equation 13 [169]. The tuning parameters that define the Kriging model are determined by maximizing the likelihood function given by Equation 14. In the worst case scenario, the likelihood function is multimodal and difficult to optimize, and a solution that is not globally optimal may yield nonsensical results [181]. However in the author's experience, the likelihood function for most problems is typically well-behaved. For this reason, a gradient-based optimizer, the

steepest descent method, will be used to train the surrogates. Computer experiments will determine whether this parameter estimation technique is appropriate or more advances tuning strategies must be employed [63, 166].

The complete Python implementation for all components of the methodology is provided in Appendix A.

4.4 Proposed Methodology

A *PFI*-based evaluation procedure is presented above along with the technical details and calculations that make up a new sampling criterion, PIC. The evaluation of competing concepts however, represents only one component of a larger framework for concept selection. A formal methodology is now presented that considers the selection task from start to finish. A flowchart of the methodology is presented in Figure 35. Details of each step follow.

4.4.1 Step 1: Define the Problem

The first and arguably most important step in any analytical process is defining the problem. Requirements are provided by the customers then key, quantifiable objectives laid out by the design engineers. Performance constraints as well as programmatic aspects are defined which ultimately guide the overall selection process.

Questions addressed during Step 1:

- What is the problem being addressed?
- What are the objectives to optimize?
- What are the performance constraints on the system?
- What is the timeframe (computational budget) for making a decision?

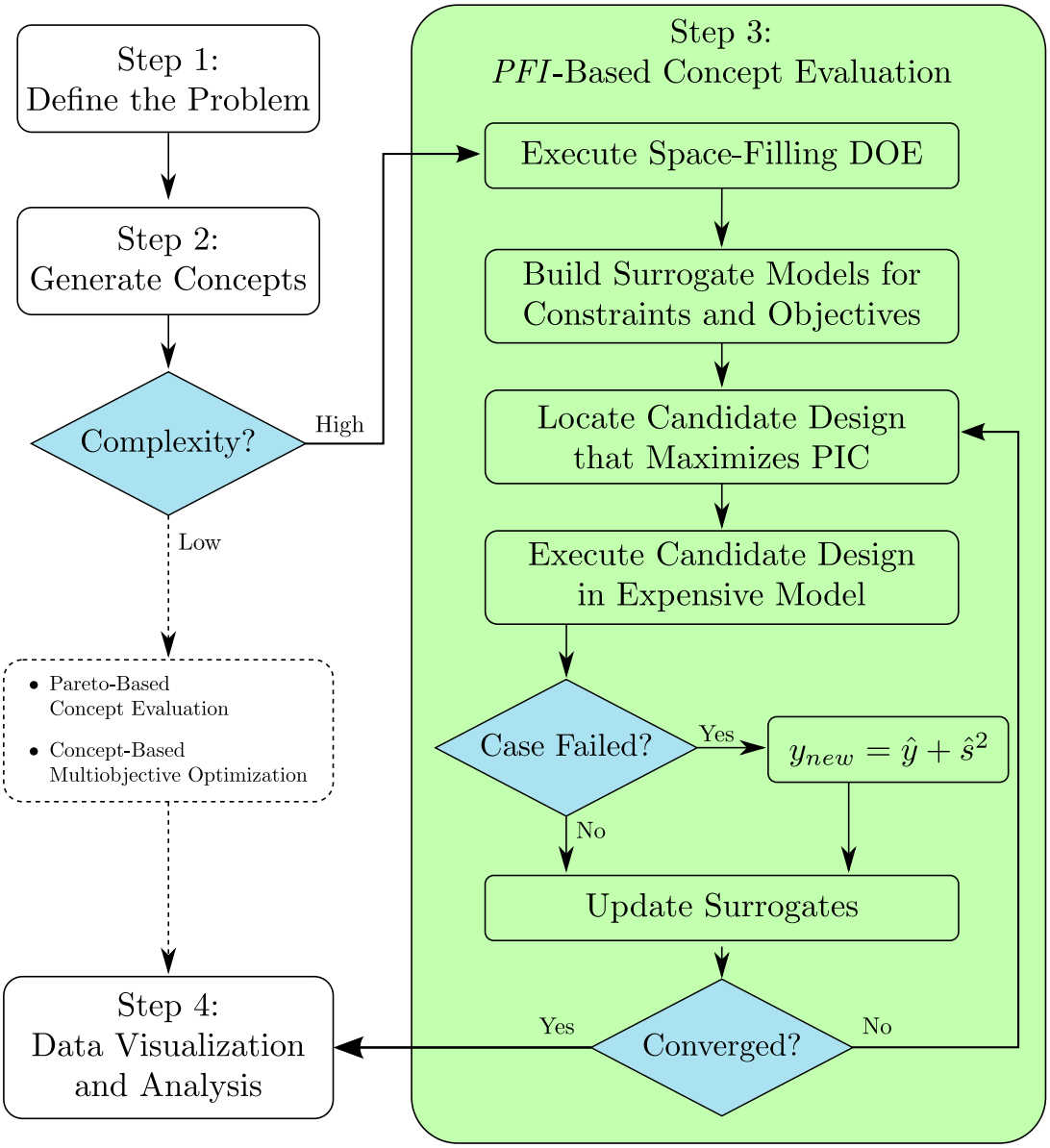


Figure 35: Proposed Methodology Flowchart

4.4.2 Step 2: Generate Concepts

The second step to a selection methodology is identification of the potential solutions to the problem. Concepts and technologies are obtained through qualitative screening, brainstorming, cross-fertilization from other disciplines/industries, morphological analysis, etc. Once the set of competing concepts is defined, the appropriate degrees of freedom and their ranges for each concept are identified. These are the design variables available to the engineer during conceptual design. Factor screening, expert opinion, Analysis of Variance (ANOVA), or other sensitivity analysis method is often utilized to pare down the design space from hundreds to a select few that drive the objectives.

Design variables are mapped to objectives during the modeling portion of Step 2. Each independent concept is represented by a computational code with unique inputs (design variables) and common outputs (objectives). Note that further sensitivity analysis may be required that makes use of computer simulation to reduce the design space.

Following concept modeling, the design engineer has a choice to make (leftmost blue box in Figure 35) which involves evaluating the complexity of the problem. The decision made at this point affects the manner in which concept evaluation is carried out. If system complexity is low (i.e. derivative concepts, evolutionary as opposed to revolutionary technologies, empirical models), a sequential or simultaneous approach to evaluation is appropriate. If models are expensive and high priority placed on each function call, the PFI-based approach must be used.

Questions addressed during Step 2:

- What concepts or technologies can satisfy the requirements?
- What are their design variables and ranges?
- How will the concepts be modeled and what is the computational complexity?

4.4.3 Step 3: *PFI*-Based Concept Evaluation

Once it has been established that *PFI*-based evaluation must be used, the procedure that is the foundation of this research begins. First an initial seeding of the space must be performed with a space-filling DOE such as Latin Hypercube. The results from the initial sample are used to train a surrogate model of all objectives and constraints for each concept. These two steps make up the initial setup of the evaluation algorithm. An iterative procedure follows whereby the space of all concepts is adaptively sampled using the PIC infill criterion to guide samples towards regions of importance. The iteration continues until some convergence criterion is met: computation budget exceeded or uncertainty reduced to a desired level. Failed cases will be handled in the manner proposed by Forrester et al. [57]. In selecting a single design of one concept that maximizes PIC, optimization of the infill criterion must be performed for every concept then compared across all concepts (maximizing the maximum). An alternative would be to run the point that maximizes PIC for each concept in the respective model at every iteration. It is assumed this does not represent any additional computational burden since concept models are independent and can be run in parallel. The same holds for retraining the surrogate model.

4.4.4 Step 4: Data Visualization and Analysis

The final step in the concept selection methodology is to visualize the multidimensional data and analysis. Results from the previous step are compiled and brought forward to provide characterization of the relationships between competing concepts. Multidimensional response data can be projected into two-dimensions via scatterplots to aid the decision maker in specifying preferences and ultimately selecting a concept. Multiobjective Pareto filters can be used to obtain the s-Pareto frontier of multiple concepts. Responses can be tied back to design variables to gain a better understanding into the behavior and relative performance of competing concepts.

Clusters of designs indicate to the designer a region where dominance shifts between concepts and may lead to further investigation so the reasons for the shift can be uncovered. The proposed Pareto Distance charts combined with a Pareto filter can be used to identify the precise location where optimality shifts between concepts and where selection decisions are most sensitive to preferences placed on the objectives.

The output of the methodology is one concept or set of concepts that are dominant as well as insight into where those concepts dominate relative to other concepts. While a final selection decision may require further modeling and simulation, poor-performing concepts may be ruled out with some degree of certainty at this stage. The designer is also provided with surrogate models of all objectives and constraints with which to make further predictions.

Questions addressed during Step 4:

- Is one concept ever preferred/dominated by another?
- What about the designs causes a shift in optimality from one concept to another?
- Where is concept selection most sensitive to preferences on the objectives?

CHAPTER V

CANONICAL PROBLEMS

Before applying the new concept evaluation methodology to UHB engine design, some preliminary experiments are performed on canonical problems. Both an algebraic sample and truss design are considered. These samples were chosen because of their ease of implementation, fast execution time, and most importantly, they are common test cases in engineering design literature for new design and optimization methods [41, 43, 102]. The proposed method must at least demonstrate the ability to solve these problems before tackling the more complex engineering design problems with expensive codes.

5.1 Experimentation Plan

In following with the scientific method, a set of low-level research questions related to method implementation are now posed. The questions are designed to address specific aspects of the algorithm's performance as well as comparison with other evaluation methods. A hypothesis is presented for each research question followed by a set of computer experiments detailing how each hypothesis will be tested according to quantifiable metrics.

5.1.1 Research Questions

Research Question 3.1: Does the PIC sampling criterion concentrate designs near Pareto frontier intersections of competing concepts?

Hypothesis 3.1: A candidate design that maximizes Pareto Intersection Closeness (PIC) will have both the property of 1) probable Pareto optimality and 2) similar

performance to designs from other concepts. When both of these properties are maximized, the design will be near the Pareto frontier intersection of competing concepts. An iterative procedure based on this sampling criterion will provide a clustering of points in this area.

Research Question 3.2: How does the PFI-based evaluation methodology compare to both sequential and simultaneous optimization approaches?

Hypothesis 3.2: Traditional quantitative methods that seek accurate Pareto frontiers are prohibitively expensive when complex analysis codes are used. PIC sampling will reduce uncertainty in areas where optimality shifts between concepts using fewer function calls than traditional design methods. These efficiency gains will allow for more expensive analysis codes to be used in early design specifically for the task of concept selection.

Research Question 3.3: How robust is the proposed method when solving less idealized problems?

For real engineering problems, the behavior of and relationship between concepts may not be as well behaved as the algebraic samples portray. There may not exist a region of intersection or their intersection may be located on the edges or corners of the design space (minimum or maximum of design variable ranges).

Another aspect of real engineering problems is that there are rarely only two objectives. While discerning dominant designs on a Pareto frontier of two dimension is straightforward, expansion into three, four, or n objectives is less trivial.

Hypothesis 3.3: PIC given by Equation 37 can be applied to problems with arbitrary

number of objectives and locates designs that are *probably* optimal whether or not a Pareto intersection exists. Mathematically, finding Pareto optimal designs in n objectives is no different than for two objectives, but the cost to find new designs (global optimization of PIC) will increase with number of objectives as well as number of current Pareto points. If concept model executions dominate overall computation time, this expense can be assumed negligible.

Visualization of results is less intuitive for problems with more than two objectives, but this is a drawback to any method and not made any more difficult with a PFI-based approach.

Research Question 3.4: What is the sensitivity of PFI-based evaluation to the assumptions aimed at increasing efficiency?

One the largest drivers in performance of adaptive sampling schemes is the appropriation of resources between initial and adaptive samples. Too few initial samples and the method could stall or waste time exploiting suboptimal regions. Kriging may also break down with too few initial points. On the other hand, too many space-filling samples could be wasteful. In order to save additional computation time, Kriging hyperparameters need not be tuned at every iteration but rather only after initial sampling. This assumption places additional emphasis on the space-filling DOE and may require an increase in initial samples to give a good fit.

Hypothesis 3.4 It is assumed, if the initial DOE is adequately space-filling, that the information needed to train can be obtained in the initial sampling of the space. This assumption is expected to hold especially for well-behaved, smooth functions where iteratively sampled points lie near previous designs (sampling that is sufficiency exploitative). Everything else being equal, a larger number of initial samples will yield a better prediction of Kriging hyperparameters. Number of initial samples will be 1)

driven down by desire to avoid wasting samples and 2) driven up by the desire to get a good initial fit. The "sweet spot" of initial samples will balance these two aspects.

5.1.2 Computer Experiments

Experiment 1: The PFI-based method will be applied to both algebraic and truss design problems and evaluated based on qualitative and quantitative metrics. Visual inspection of the sampling will be performed for the two-objective problems where relationships between concepts can be easily deduced. A qualitative measure of performance will be the algorithm's ability to reduce predictor uncertainty at known locations in the design space. Kriging and other Bayesian models provide a statistical interpretation of uncertainty that reflects the lack of knowledge about an unexplored area of an expensive code. In selecting between concepts, designers would like to reduce this lack of knowledge particularly in regions where decisions carry the most risk. Kriging error (given by Equation 19) as a function of iteration number will be compared at four locations: 1) Pareto intersection (found analytically), 2) optimal for a single concept, 3) s-Pareto optimal, and 4) suboptimal.

Experiment 2: The method will be compared to representative methods from each class of techniques: sequential (multiobjective Probability of Improvement, NSGA-II, DOE) and simultaneous (multi-Pareto Probability of Improvement). Visual inspection of the sampling as well as reduction in predictor uncertainty will be compared for both alternatives to PFI-based evaluation from Experiment 1.

Experiment 3.1: The initial formulation of the algebraic sample contains two concepts with a single point of Pareto intersection. This experiment will investigate the performance of the method when this relationship is not as ideal. Two cases will be considered. First, the concepts will be artificially shifted such that all designs that are dominant for a single concept are also on the s-Pareto frontier. That is, the Pareto frontiers of both concepts completely describe the s-Pareto frontier. The second case

will look into the situation where one concept completely dominates the other. These aspects are all contained in the truss design problem where six truss concepts occupy the objective space.

Experiment 3.2: The algebraic sample will be scaled to three and four objectives according to Deb [43]. Results will be projected into two dimensional scatter-plots to infer regions where dominance shifts between concepts. A quantitative metric will be Euclidean distance from each iteratively placed sample to the Pareto frontier of the competing concepts. This metric can be plotted as a function of the objectives as well as design variables to deduce locations of optimality shifts between concepts in the multiobjective space.

Experiment 3.3: Computation time required to locate adaptive samples will be tracked at each iteration. The data will be used to develop relationships between number of samples, number of Pareto points, and number of objectives.

Experiment 4.1: For a fixed budget of function calls, the ratio of initial to adaptive samples will be varied for the algebraic sample problem. Reduction in uncertainty at the known intersection will be compared for various sampling splits. The relationship between number of initial samples and degrees of freedom will be used to establish an upper bound on number of required space-filling points.

Experiment 4.2: Kriging hyperparameters will be tuned at every iteration and tracked for various sampling splits similar to those used in Experiment 4.1. The point at which the hyperparameters become independent to number of initial samples will provide a lower bound for number or required space-filling points.

5.2 Algebraic Sample Problem

The first implementation of the proposed evaluation method is an algebraic sample problem consisting of two concepts (A and B) with two design variables (x_1 and x_2) and two objectives (f_1 and f_2). The algebraic sample is a modification to Deb's

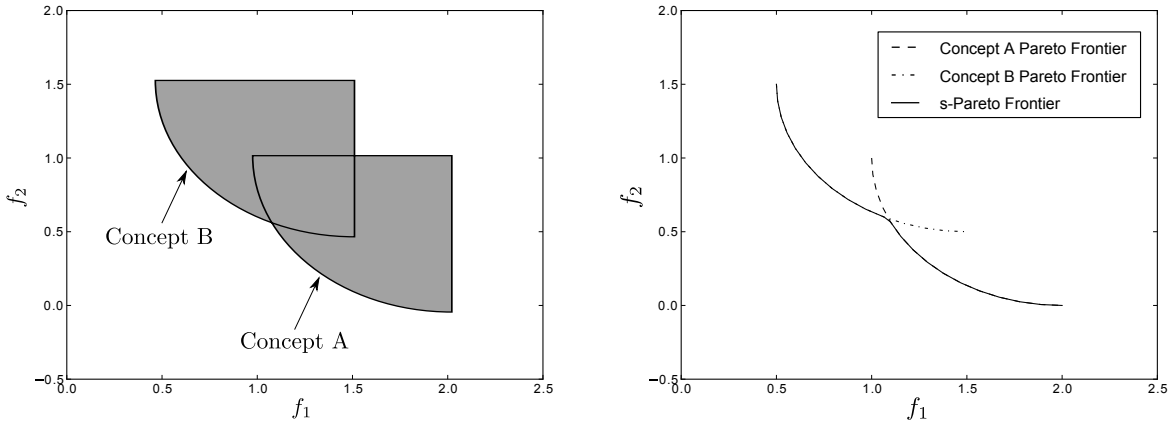
scalable sphere problem used for testing multiobjective evolutionary algorithms [43]. The two objectives are defined by Equations 38 and 39.

$$f_1^{(k)}(x) = -(1 - x_1^2)cos(x_2) + M^{(k)} \tag{38}$$

$$f_2^{(k)}(x) = -(1 - x_1^2)sin(x_2) + N^{(k)} \tag{39}$$

$$\text{Subject to:} \quad -1 < x_1 < 1, \quad 0 < x_2 < \frac{\pi}{2}$$

where $f^{(k)}$ is the objective for the k^{th} concept, $[M^{(A)}, N^{(A)}]$ is $[2.0, 1.0]$, and $[M^{(B)}, N^{(B)}]$ is $[1.5, 1.5]$. Both objectives are minimized when $x_1 = 0$. The concepts are overlapping quarter-circles depicted in Figure 36. Figure 36a shows the overlapping objective spaces. Figure 36b shows the individual Pareto frontiers of each concept along with the s-Pareto frontier of the multi-concept space. Both concepts have portions of their own frontiers that are also s-Pareto dominant.



(a) Multi-Concept Design Space

(b) Pareto Frontiers

Figure 36: Algebraic Sample Problem

5.2.1 Experiment 1: Method Performance

Each concept was initially seeded with ten designs followed by 50 adaptive samples choosing the design that maximized PIC at each iteration. The initial/adaptive sample split will be explored in later experiments, but it is assumed here that number of

initial samples be five times the number of design variables. The initial sample was a space-filling Latin Hypercube DOE. The maximum iteration count of 50, while arbitrary, was chosen as the point at which error (shown below) seem to 'taper off' and become somewhat constant owing to numerical precision in the Kriging predictor. At each iteration, a sample was selected for both concepts requiring two separate global optimizations of PIC, a highly multimodal function. A genetic algorithm (GA) with tournament selection and elitism was used to find the candidate that maximized the infill criterion. The GA was implemented with a population of 50 individuals evolved over five generations for a total of 250 calls of each Kriging model.

Figure 37 shows both objectives with initial and adaptive samples for Concepts A and B.

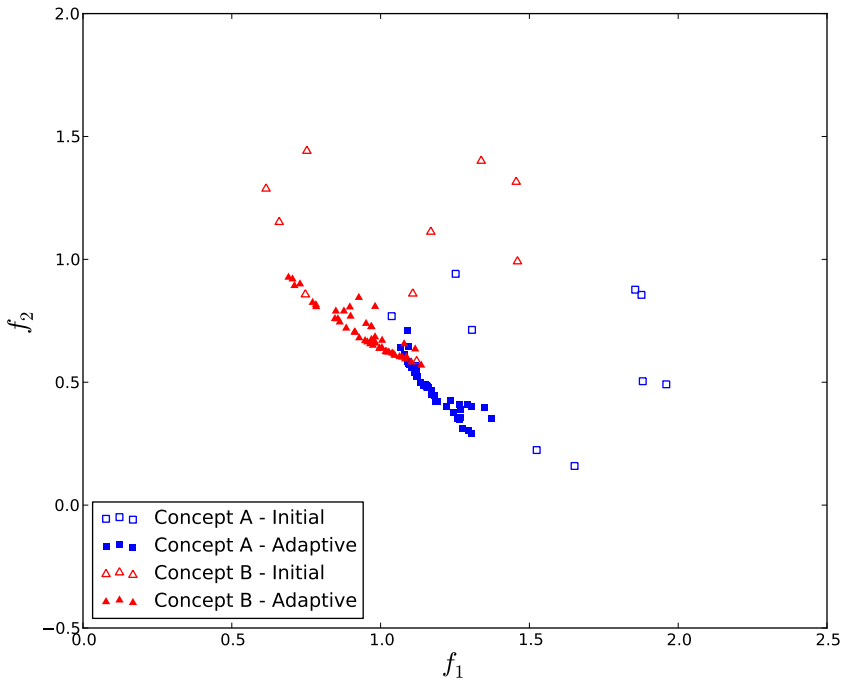
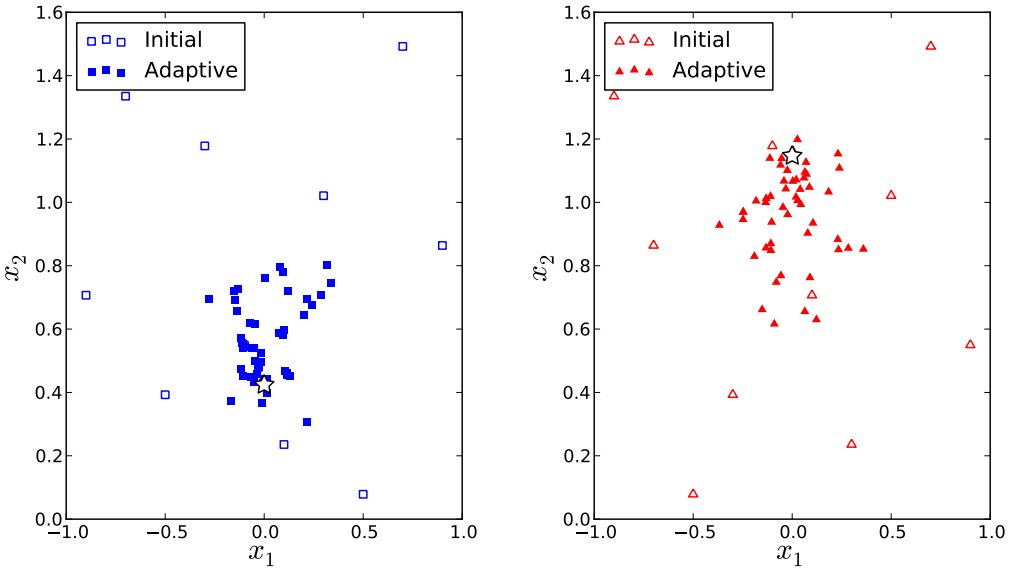


Figure 37: PFI-Based Sampling for Algebraic Sample Problem

The initial, space-filling samples are shown by open shapes while iterative samples shown by closed shapes. After 60 model executions, the intersection between Concept

A and B is modeled quite well. Notice the extreme corners of the s-Pareto frontier were ignored by the iteration. This is not due to the fact that these areas are not important, but rather enough is understood about this area to know that another concept dominating here is highly improbable. The information gained from the initial sampling was enough to determine that running more samples there was unnecessary.

The design variables corresponding to the iteration are plotted in Figure 38. The true intersections, found analytically, are depicted as stars in the design space. In contrast to DOE methods where a uniform sampling results (open shapes), clustering of adaptively sampled designs (closed shapes) can be seen in areas of intersection. Another desirable property of the sampling is that samples, while close to the true intersection are somewhat biased towards Pareto optimality and away from where the competing concept dominates.



(a) Concept A

(b) Concept B

Figure 38: Design Variable Clustering with PFI-Based Sampling

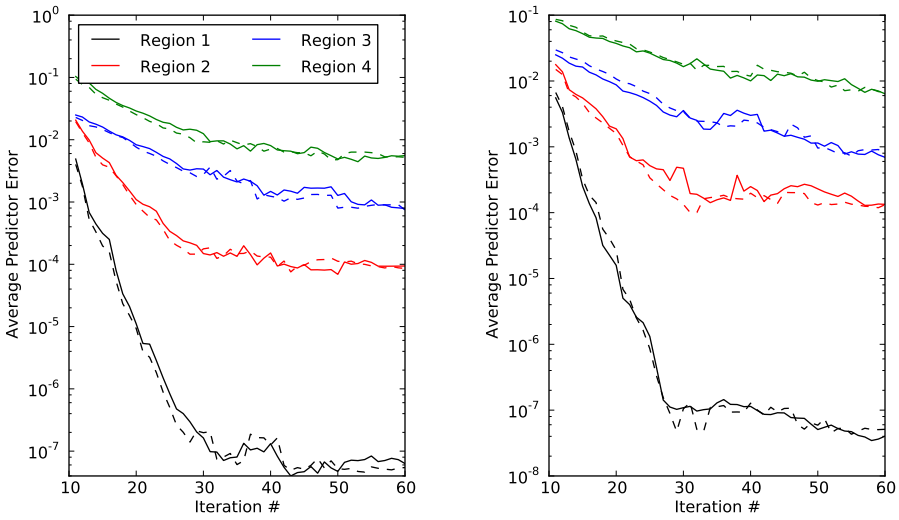
A quantitative measure of the algorithm's sampling efficiency is now presented.

Leveraging the Kriging surrogate for each concept, iteration history of predictor error (given by Equation 19) is compared at four unique locations in the objective space. The four points are defined in Table 10. Designs in Regions 2, 3 and 4, while somewhat arbitrary, were chosen as representative of a particular region of the design space.

Table 10: Regions for Quantitative Comparison

Region	Description	$x^{(A)}$	$x^{(B)}$
1	Pareto intersection	[0.0,0.42403]	[0.0,0.14677]
2	Optimal for a single concept	[0.0,0.0]	$[0.0,\frac{\pi}{2}]$
3	s-Pareto optimal	$[0.0,\frac{\pi}{2}]$	[0.0,0.0]
4	Suboptimal	[1,0]	[1,0]

Due to the randomness of the initial Latin Hypercube sampling, the iteration was repeated ten times and results averaged over all ten experiments. Repetition of the iteration was meant to eliminate any chance that the initial seeding of the design space placed a point close to the Pareto intersection. This way, a reduction in predictor error is assumed to represent a clustering of samples, not merely "getting lucky." The iteration history for the four regions is shown in Figure 39. The first and second objectives are shown by solid and dashed lines respectively.



(a) Concept A (b) Concept B

Figure 39: Iteration History for PFI-Based Sampling

The black line representing predictor error at the Pareto intersection sees the largest decrease. The clustering in this region reduced the error significantly relative to the other three regions. As expected, the suboptimal areas (green line) remain the least sampled and highest error throughout the iteration for both concepts. The error in Region 2 is less than Region 3 due to the proximity of the representative design to the true intersection even though a majority of the samples are dominant.

5.2.2 Experiment 2: Method Comparison

The above experiment is now repeated using MOPI and MPPI as sampling criteria as representative sequential and simultaneous optimization approaches respectively.

5.2.2.1 MOPI

Objective space and design variable sampling are shown in Figures 40 and 41.

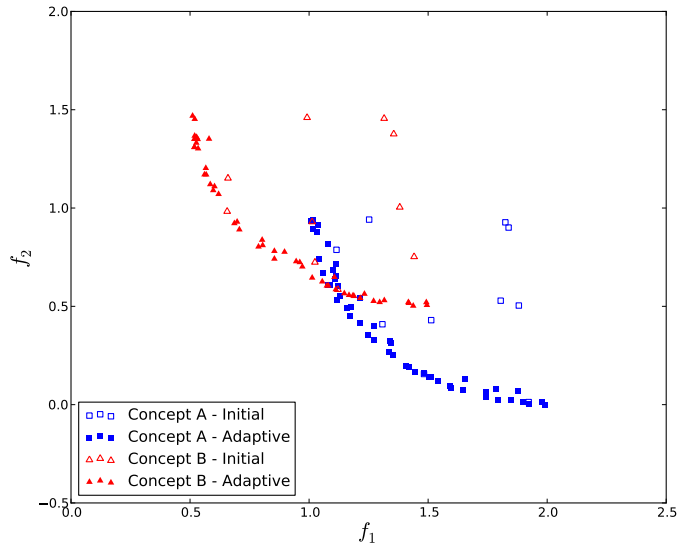
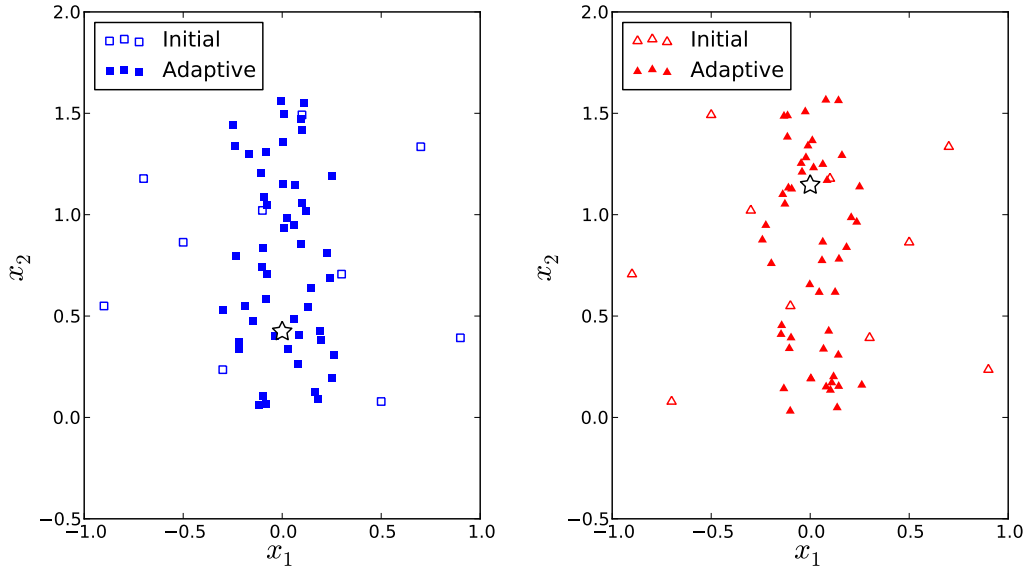


Figure 40: MOPI-Based Sampling for Algebraic Sample Problem

The sequential method is successful at locating the Pareto frontiers of both concepts very accurately. Concept evaluation is performed in isolation which leads to some samples being placed on Pareto frontiers of individual concepts but otherwise dominated. Sampling of the design space only confirms this as adaptive points are placed all along the s-Pareto frontier where $x_1 = 0$.

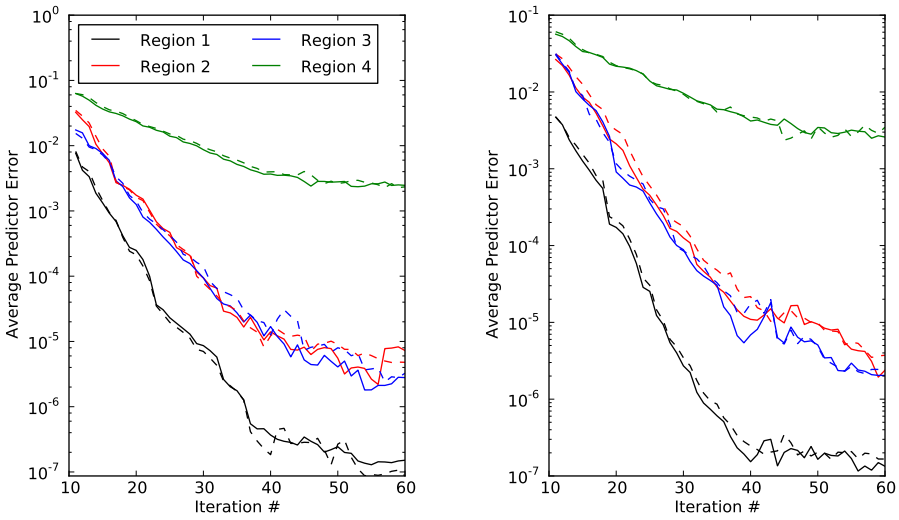


(a) Concept A

(b) Concept B

Figure 41: Design Variable Clustering with MOPI

Figure 42 shows the iteration history of uncertainty for MOPI iterations. The iteration shows reduction in intersection error relative to other regions, but this occurs after many more function calls than for PIC for the same level of uncertainty. The location of the intersection relative to the corners of the design space contributes to the reduction in error for this sampling. The sampling also makes no distinction between s-Pareto and dominated regions as seen by the similar reduction in error of the blue and red lines.



(a) Concept A

(b) Concept B

Figure 42: Iteration History for MOPI-Based Sampling

5.2.2.2 MPPI

Sampling based on MPPI is shown in Figures 43 and 44. The s-Pareto frontier is modeled accurately, even at the edges of the space where a better concept is highly unlikely. Dominated regions are avoided because sampling is now influenced by performance of other concepts. In the design space sampling, samples are clearly clustered about the region where $x_1 = 0$ and stop at the intersection indicating that only designs along the s-Pareto frontier are desirable. This is more pronounced for Concept B (red triangles) as no adaptive samples were placed in the dominated region. Five executions of Concept A were performed that were dominated by Concept B. However, these samples were all very close to the s-Pareto frontier and can be explained by two things: 1) model uncertainty in the Kriging leading to explorative search or more likely 2) PI is merely an approximation of true probability of improvement found by Monte Carlo simulation.

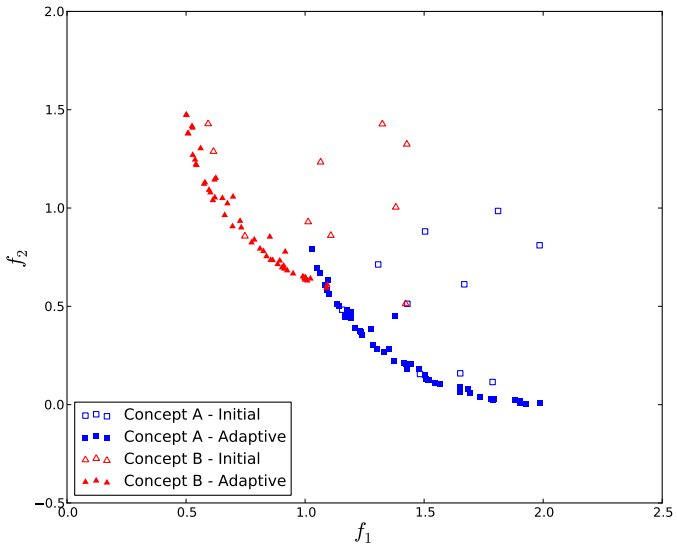
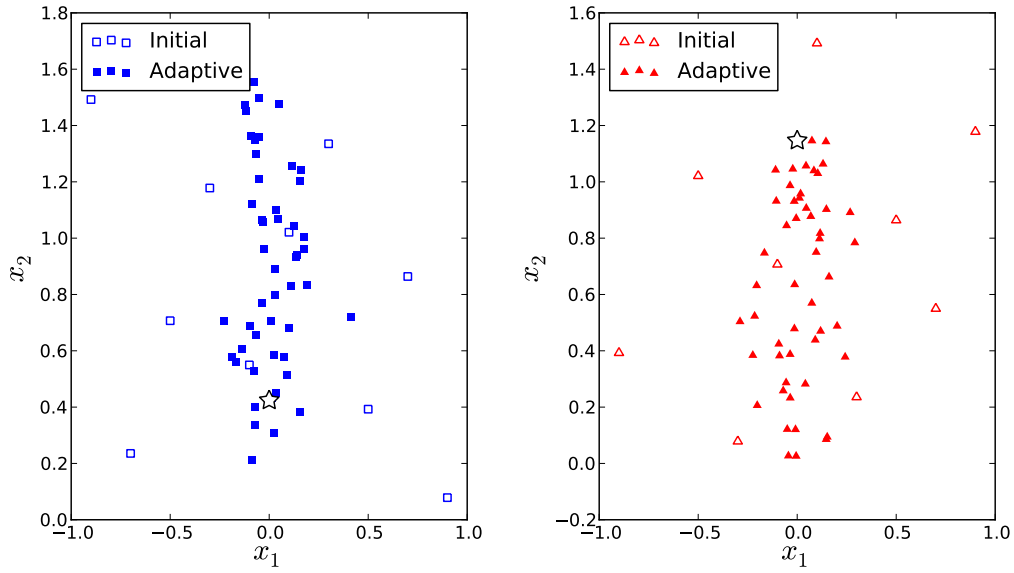


Figure 43: MPPI-Based Sampling for Algebraic Sample Problem



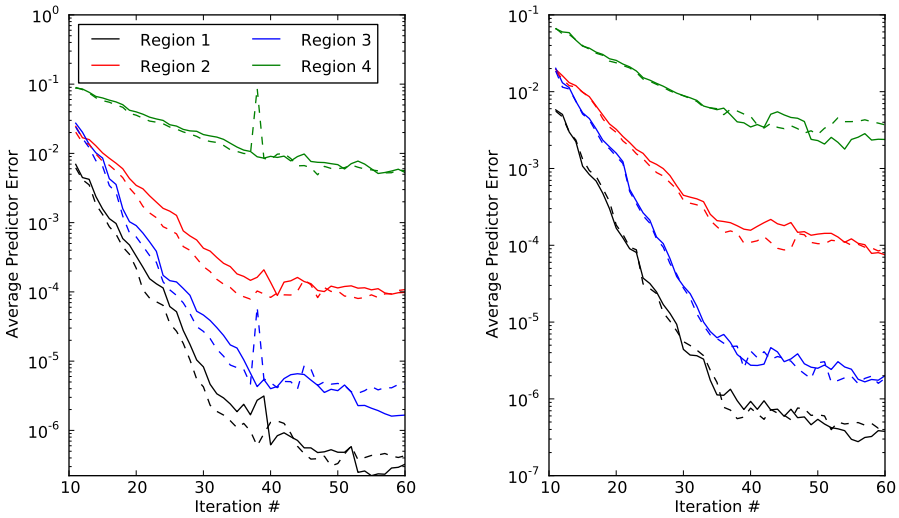
(a) Concept A

(b) Concept B

Figure 44: Design Variable Clustering with MPPI

Iteration history in Figure 45 shows higher uncertainty about the intersection

but much lower in areas of strict s-Pareto dominance compared with PIC sampling. Suboptimal areas still have high uncertainty relative to other areas which is desirable. The sampling of Pareto optimal designs is still too dense as these points only confirm what is already known about concept dominance. When function calls are at a premium, this is a waste of resources.



(a) Concept A

(b) Concept B

Figure 45: Iteration History for MPPI-Based Sampling

5.2.2.3 NPD

For comparison, the experiment was repeated using NPD as a sampling criterion. Sampling based on NPD is purely exploitative. That is, designs are located that minimize the distance to *existing* dominant points of other concepts - what is already known about the space. This leads to dense clusters of designs whose locations are highly dependent on initial sampling of the space. Notice the large collection of triangles surrounding the single initial square sample in the zoomed in portion of Figure 46. This initial sample was placed for Concept A in an area that was dominated by Concept B. Rather than push towards more optimal, a handful of samples were

placed about this point. This behavior is explained by examining Equation 31. It is very easy for Concept B to achieve designs near here as it falls well within its own design space. More simply, Concept B will only try to minimize distance of its own designs to existing designs of other concepts. While the purely exploitative property may lead to fast convergence, with no way to explore potentially Pareto optimal areas, samples may "get stuck" near suboptimal designs.

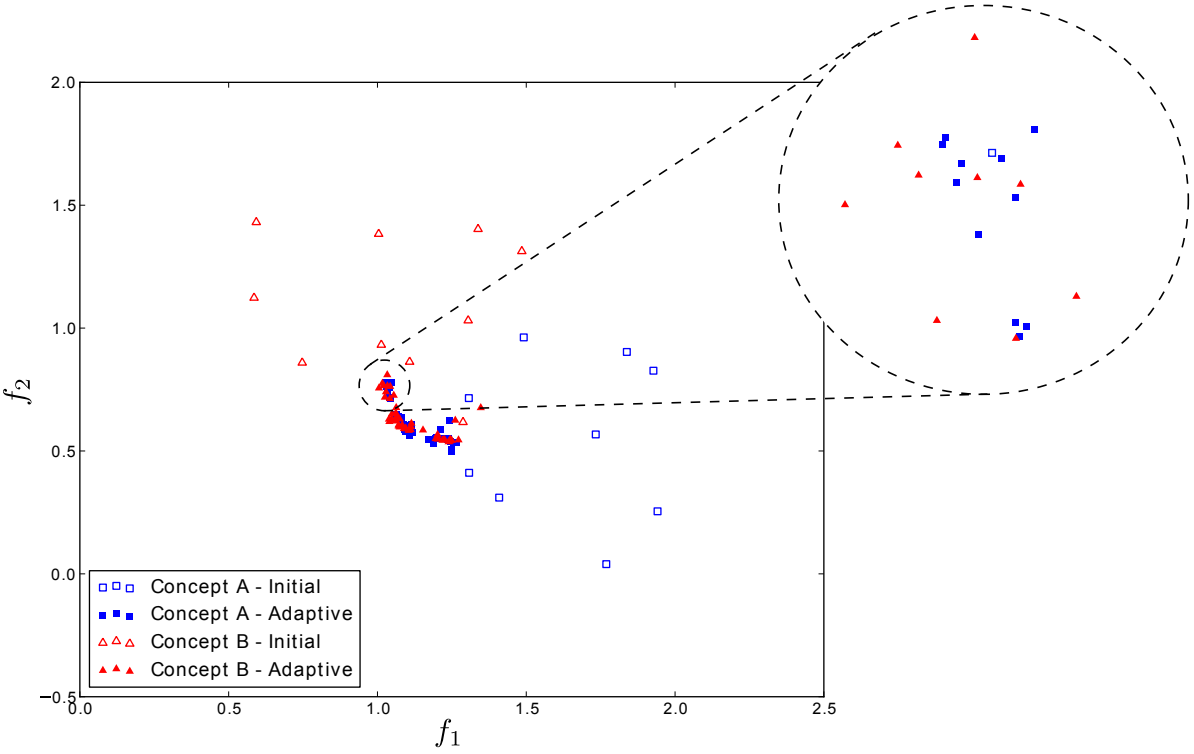


Figure 46: NPD-Based Sampling for Algebraic Sample Problem

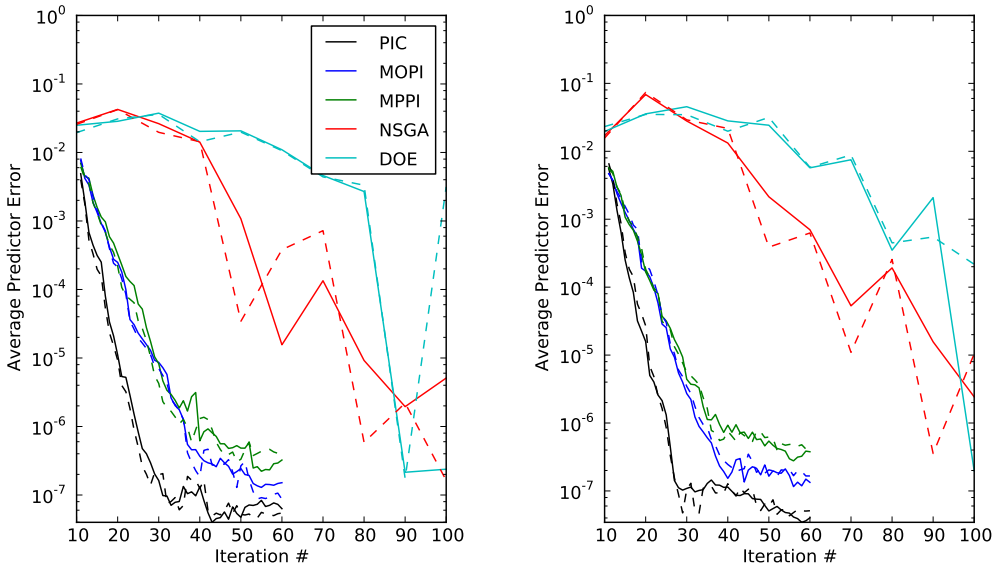
5.2.2.4 Zero-Overhead Methods

The representative methods used in the experiments above all rely on adaptive sampling to gradually increase prediction accuracy. The results are obtained by essentially substituting the appropriate infill criterion for each experiment. This allowed direct comparison between methods where the overall procedure is identical: 1) execute initial sample, 2) train surrogate models, and 3) locate designs that optimize infill criterion. The experiments are now repeated using "two-stage" methods where the

entire set of samples is generated first then surrogates fit through the results. These methods do not have the additional "overhead" cost of optimizing an infill criterion to locate additional designs. Two techniques for generating the sample sets will be used: DOE and NSGA-II. While one could argue that NSGA-II is a form of adaptive sampling, these methods differ from the above approaches in that total computational cost is purely a function of model executions. The cost of genetic algorithms operators (crossover, mutation, and selection) are assumed negligible. A Latin Hypercube DOE was chosen due to its pervasiveness in the engineering community and ability to space designs evenly throughout the design space. Without *a priori* knowledge of system behavior, the most space-filling DOE is desired. NSGA-II was chosen as a representative multiobjective evolutionary algorithm and one of the most widely used MOO techniques. NSGA-II was also used to benchmark against the analysis by Berton et al. in [23].

Kriging models were built around the final set of available data and predictor error calculated for the four regions. The resulting sampling and iteration histories for each of these two methods are given in Appendix C.1. The two-stage methods are now compared with adaptive sampling for Region 1. The iteration histories of all methods are shown together in Figure 47. Due to poor performance of the DOE and NSGA-II, total number of samples was increased to 100 (in increments of 10) for comparison with adaptive sampling. The jaggedness of the iteration histories of these two methods indicates that DOE and NSGA are much more sensitive to sample randomness. While repetition of the experiment may smooth out this behavior, the general trend is obvious and highlights a major drawback that these two-stage approaches require too many function calls. In moving to a surrogate-based adaptive sampling scheme, performance is increased drastically with the most accurate prediction coming from the proposed PFI-based approach. This result is not surprising as PIC is the only sampling criterion to date *designed* to concentrate samples into

Pareto frontier intersections and enable PFI-based evaluation.



(a) Concept A

(b) Concept B

Figure 47: Uncertainty Reduction in Region 1

5.2.3 Experiment 3: Robustness

The purpose of the following experiments is to investigate the performance of the PFI-based evaluation methodology when the problems are less than ideal. First, the relationship between concepts is changed such that there is no longer a well behaved intersection. Second, the performance of the method is investigated for problems with more than two objectives.

5.2.3.1 Experiment 3.1: No Intersection

Case 1: Concepts are artificially shifted apart such that there are no designs that are dominated by other concepts. In other words, all Pareto points from the respective concepts fall on the s-Pareto frontier. The shift parameters are defined as $[M^{(A)}, N^{(A)}]$ equal to $[3.0, 1.0]$, and $[M^{(B)}, N^{(B)}]$ is $[1.0, 3.0]$. The resulting objective space is depicted in Figure 48. While there is a region where optimality shifts between concepts,

there is no true intersection and a discontinuous s-Pareto frontier. Adding to the difficulty of the problem is the fact that the location where a shift exists is on the edge of the design space for x_2.

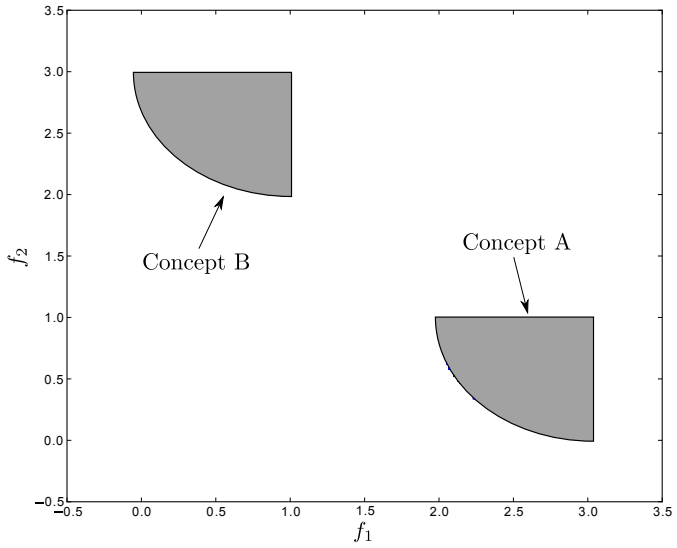


Figure 48: Two Concepts with no Pareto Intersection

The experiment is repeated using the same setup as with Experiment 1 (ten initial samples and 50 adaptive). Sampling results are shown in Figure 49 and corresponding design variable in Figure 50. Inspection of the sampling reveals that in fact, iterative designs are placed where optimality shifts between concepts. Designs are clustered on the s-Pareto frontier in areas that are closer to the competing concept.

Another desirable feature is the ability to locate designs in the corners or edges of the design space. This is seen in Figure 50. Designs are clustered towards the bottom of the range of x_2 for Concept A and top of the range x_2 for Concept B.

Case 2: Concepts are shifted such that one concept completely dominates the other. This is done using the following shift parameters: $[M^{(A)}, N^{(A)}]$ is $[2.0, 1.0]$, and $[M^{(B)}, N^{(B)}]$ is $[3.0, 2.0]$. The individual Pareto frontier of Concept A completely describes the s-Pareto frontier whereas no designs from Concept B are dominant.

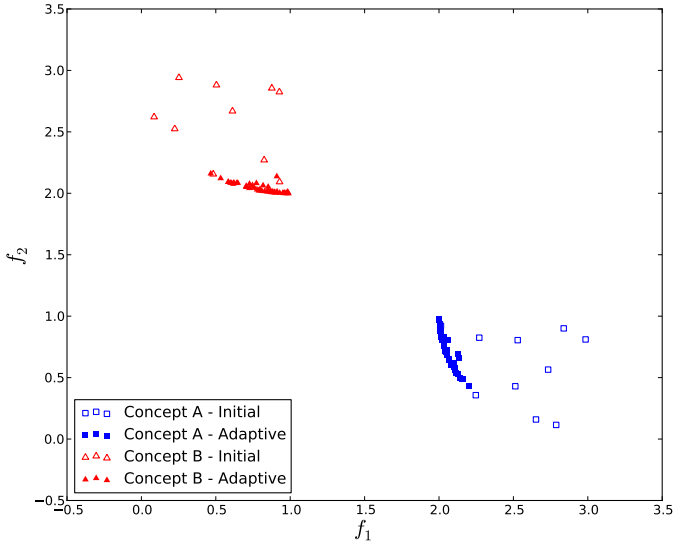
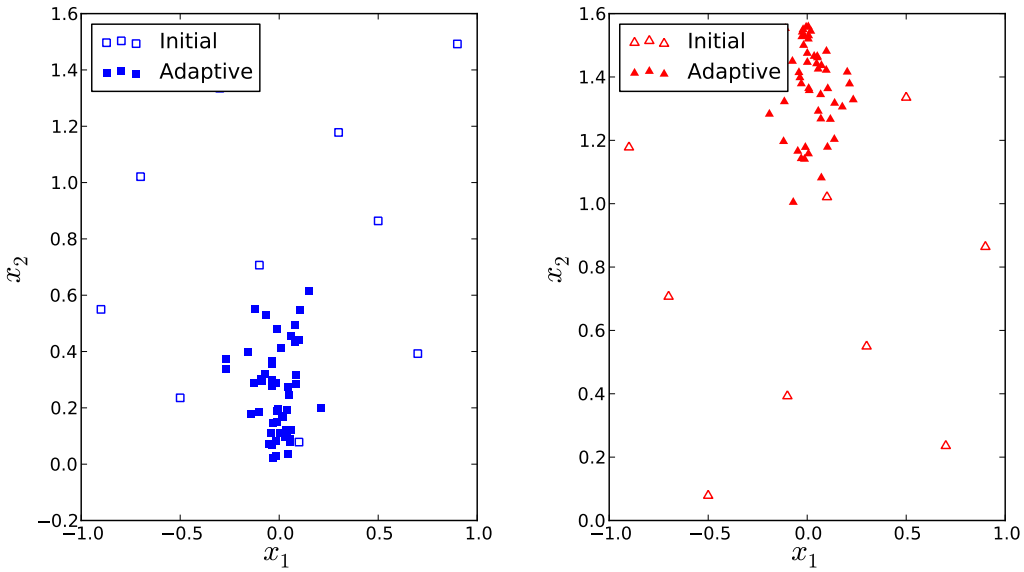


Figure 49: PIC-Based Sampling for Concepts with no Intersection



(a) Concept A

(b) Concept B

Figure 50: Design Variable Sampling for Concepts with no Intersection

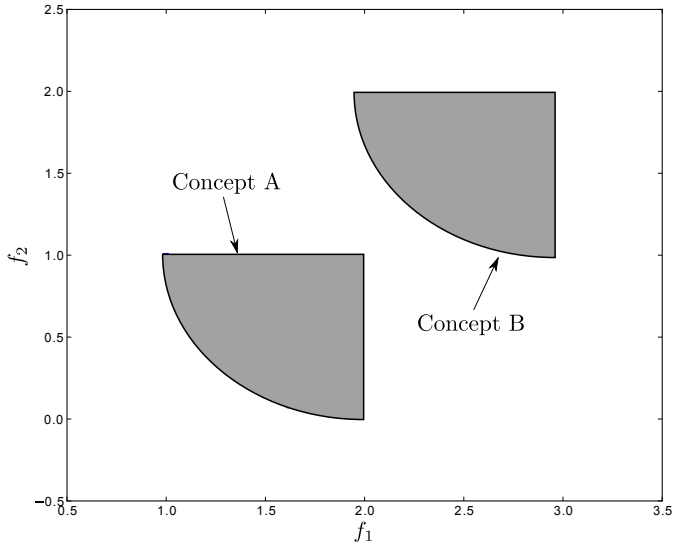


Figure 51: One Concept Completely Dominant

This is depicted in Figure 51.

The resulting sampling is shown in Figure 52 and is a much more interesting scenario than Case 1. Sampling of the dominated concept (red points) appears to be distributed along the individual Pareto frontier of Concept B but biased towards the center of the frontier. This result can be explained by examining the formulation of PIC. If the current concept is never dominant, the search reduces to finding those designs that are closest to Pareto points of the other concept. The probability of improving on the current set of designs, MPPI, is very small, so NPD dominates the infill criterion. This is not as troubling as it may seem, because any point that minimizes distance to a dominant concept's Pareto frontier, will also be optimal for that concept.

The opposite situation occurs for concepts that are entirely dominant. In this example, the s-Pareto frontier formed without Concept A designs is an empty set (no designs from Concept B). PIC reduces to only MPPI and search progresses for evenly spaced Pareto designs.

Design variable clustering is seen in Figure 53 and again highlights the emphasis

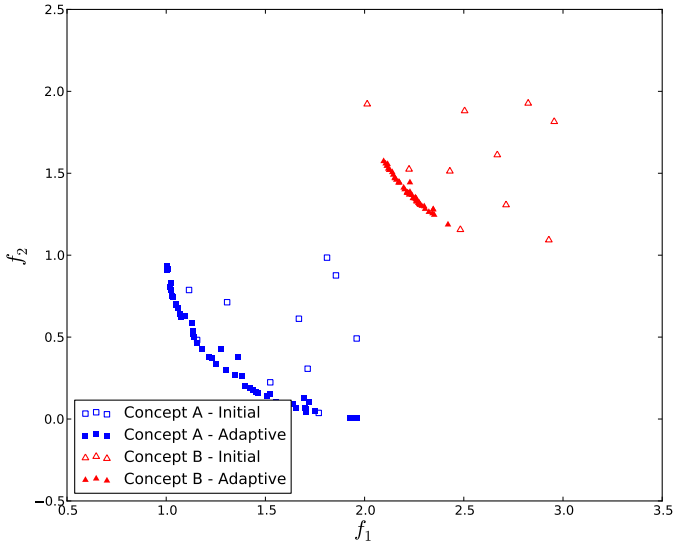


Figure 52: PIC-Based Sampling for Completely Dominant/Dominated Concept

on MPPI for Concept A (Figure 53b). Samples are placed all along the acceptable range of x_2 while being centered around $x_1 = 0$ indicating an even spacing of Pareto optimal (or near optimal) points. The exploitative nature of NPD is shown in Figure 53b. The search is less concerned with optimality but nevertheless evaluates points that are optimal because they are close to Concept A.

5.2.3.2 Experiment 3.2.1: Increased Number of Objectives

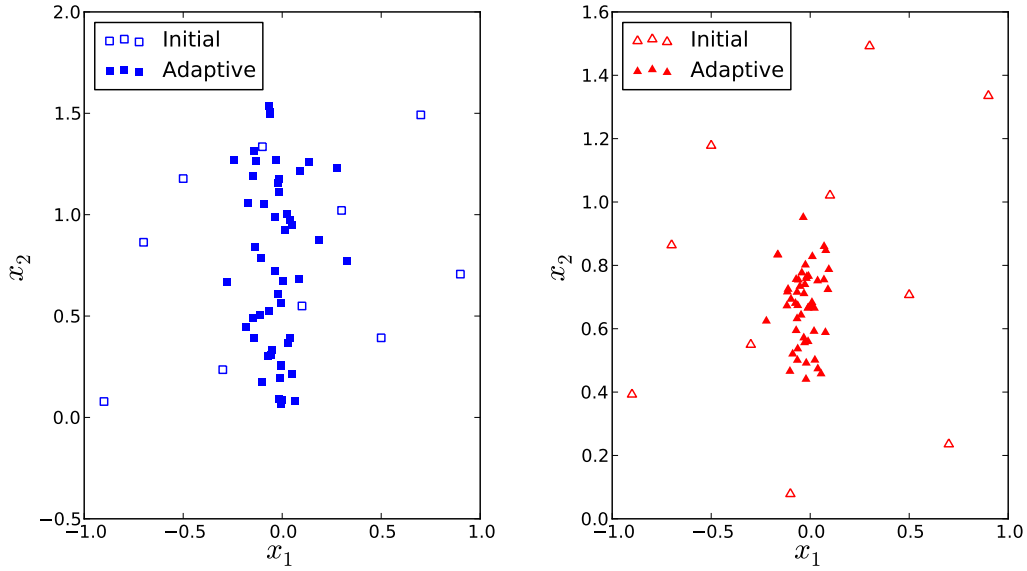
The algebraic sample problem is now expanded into three objectives to investigate robustness of the method to increased number of objectives. The three objectives for each concept are given by Equations 44-46.

$$f_1^{(k)}(x) = -(1 - x_1^2)cos(x_2) + M^{(k)} \quad (40)$$

$$f_2^{(k)}(x) = -(1 - x_1^2)sin(x_2)cos(x_3) + N^{(k)} \quad (41)$$

$$f_3^{(k)}(x) = -(1 - x_1^2)sin(x_2)sin(x_3) + O^{(k)} \quad (42)$$

$$\text{Subject to:} \qquad -1 < x_1 < 1, \quad 0 < x_{2,3} < \frac{\pi}{2}$$



(a) Concept A (b) Concept B

Figure 53: Design Variable Sampling for Completely Dominant/Dominated Concept

where $[M^{(A)}, N^{(A)}, O^{(A)}]$ is $[2.0, 1.0, 1.0]$, and $[M^{(B)}, N^{(B)}, O^{(B)}]$ is $[1.5, 1.5, 1.5]$. The three-dimensional representation of the objective space is shown in Figure 54. The two concepts, when scaled to three objectives, become overlapping one-eighth sphere sections. The intersection is no longer a single point but a line of potential designs with similar performance. For problems with three objectives, visualization can be performed in the manner presented in Figure 54 (higher dimensional problems will be discussed later). Two-dimensional projections can further aid in locating intersections, but only insofar as the intersections are aligned with the objective axes. Lines or planes of Pareto intersection that are located arbitrarily in the objective hyperspace can be impossible to identify in two dimensions. For this reason, a visualization technique is presented in later sections to overcome this difficulty. As will be shown in subsequent examples, the technique can be used for problems with greater than three objectives.

Continuing with the sphere test problem, a Monte Carlo sampling of the space is

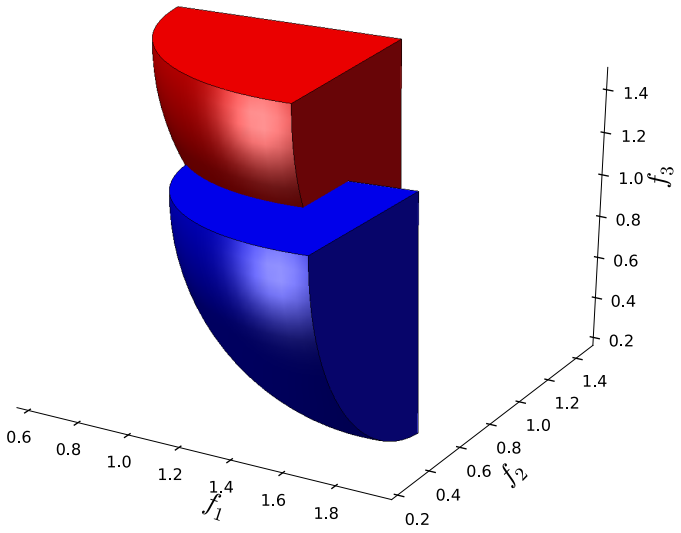


Figure 54: Three-Dimensional Representation for Algebraic Sample

presented in two-dimensional projections in Figure 55. The projection of f_2 versus f_3 shows Concept A entirely dominating when in fact there is a tradeoff between A and B in the other objectives. This highlights an additional difficulty in visualizing Pareto hypersurfaces where even Pareto optimal designs appear suboptimal.

The sampling was performed for 50 iterations as in Experiment 1 and plotted in Figure 56. The initial sampling was again assumed to be five times the number of design variables (15) yielding a total of 65 function calls to each concept. The projection of f_1 versus f_2 and f_3 are relatively similar in behavior to the results obtained for the two-objective case. However, the upper right portion of the figure (f_2 vs. f_3) shows a heavy concentration of designs in a region of apparent suboptimality. While they appear to be suboptimal in this projection, they are in fact on the s-Pareto frontier. The samples are placed here because of the strong possibility that a Pareto intersection exists and the low probability that Concept B dominates Concept A along these two objectives.

The sampling results are shown in three dimensions in Figures 57 and 58. The initial sample is removed for clarity and adaptive samples are plotted as blue and red

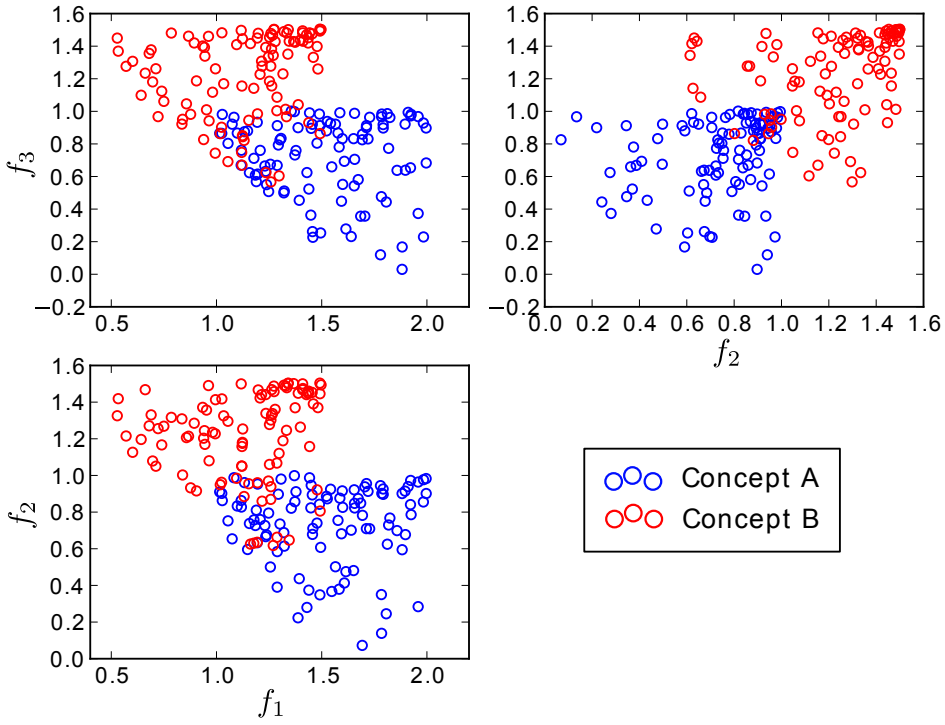


Figure 55: Two-Dimensional Projections for Algebraic Sample

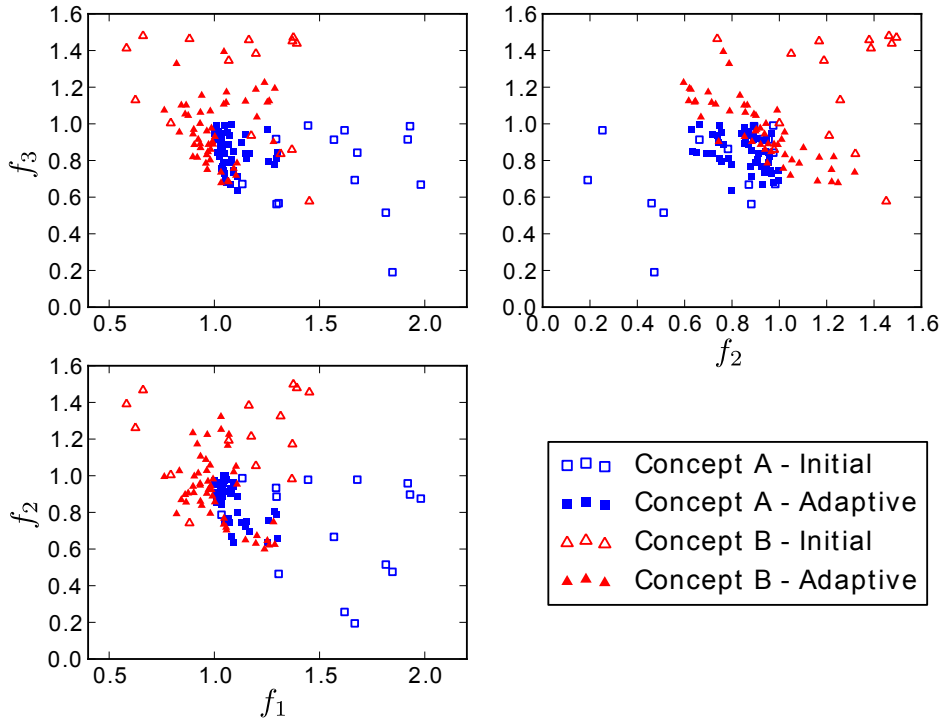
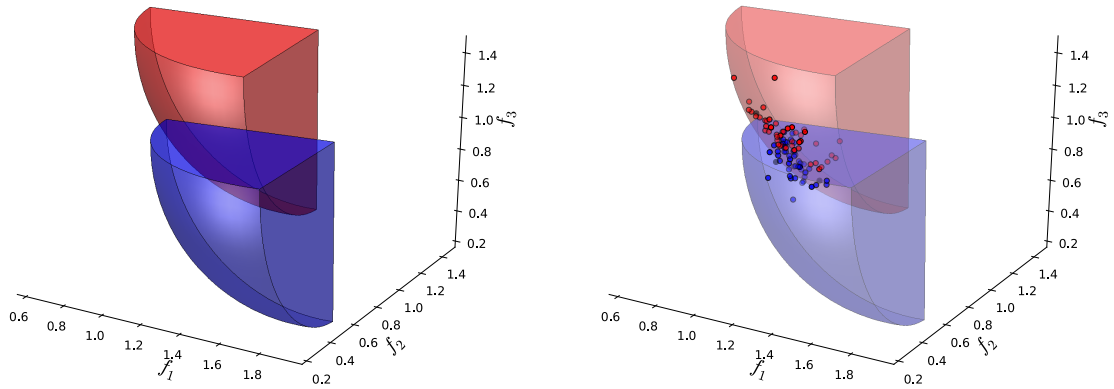
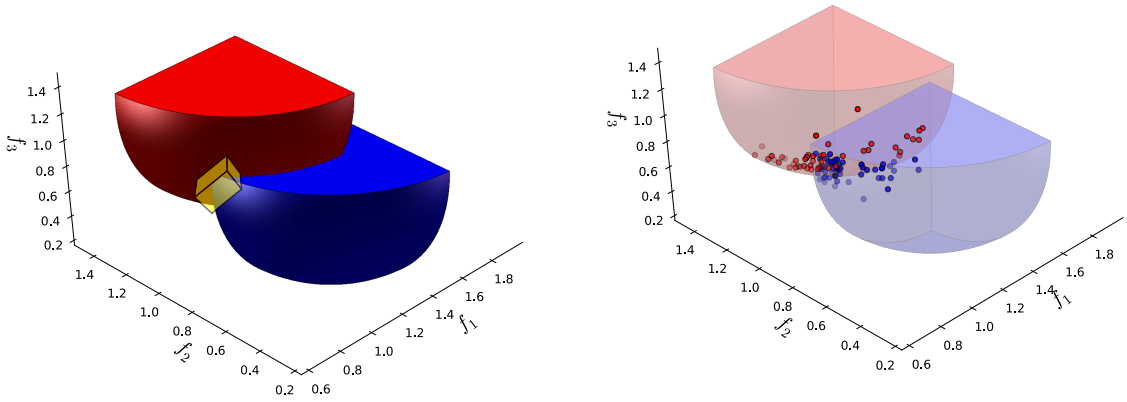


Figure 56: PFI-Based Sampling in Three Objectives



(a) Overlapping Concepts (b) Sampling

Figure 57: Three Dimensional View of Sample Clustering - Projection 1



(a) Overlapping Concepts (b) Sampling

Figure 58: Three Dimensional View of Sample Clustering - Projection 2

for Concept A and B respectively. Two perspectives are shown to depict the three dimensions more clearly. The precise intersection of the sphere sections is in fact a line of designs highlighted in Figure 58a. The samples are shown to be clustered around this area.

For problems with more than three objectives, visualization becomes very difficult. The novel technique is proposed for presenting results obtained from the PFI-based evaluation called the Pareto Distance Chart. A quantitative measure that will be used to identify optimality shift is Euclidean distance of samples to competing concepts'

Pareto frontier. Pareto Distance, PD of sample y^A from Concept A is given by Equation 43.

$$\text{PD} = \min \sqrt{\sum_{i=1}^{n} (\bar{y}_{i,j}^B - y_i^A)^2}, \quad j = (1, 2, ..., M) \tag{43}$$

where $\bar{y}_{i,j}^B$ is the j^{th} Pareto optimal design along the i^{th} objective from Concept B.

The proposed Pareto Distance Chart concept is shown as a function of the objectives in Figure 59. A value of zero indicates a sample is placed directly on top of a Pareto optimal point of the other concept. As expected, the space-filling points (empty shapes) show the largest PD, while the iteratively sampled points are clustered near zero. The triangular shape to the data with a sharp point near $PD = 0$ indicates a single point of Pareto intersection along the particular objective. This can be seen in the leftmost plot of Figure 59. A more shallow, "bowl-shaped" behavior indicates a larger region where two concepts have similar performance (rightmost plot in Figure 59). This indicates a line (or plane or hyperplane) of intersection where similar performance between concepts can be achieved over a range of objective values. Iterative designs (closed shapes) that radiate outward from zero are still optimal but more unlike designs of competing concepts and farther from the intersection. To identify the values of the objectives where preferences are most important, a decision maker can use the Pareto Distance Charts as a two-dimensional representation of the data. In this example, optimality shifts between concepts at $f_1 = 1.0$. As preference on the first objective increases, Concept B becomes more desirable as evidenced by the clusters of samples (red triangles) to the left of $f_1 = 1.0$ in Figure 60. Increased preference on the second and third objectives move desirability towards Concept A.

Pareto Distance is plotted as a function of the design variables in Figure 61. A clustering of designs on either side of a design variable shows exactly where in the design space a concept dominates. An example of this is the middle plot of Figure

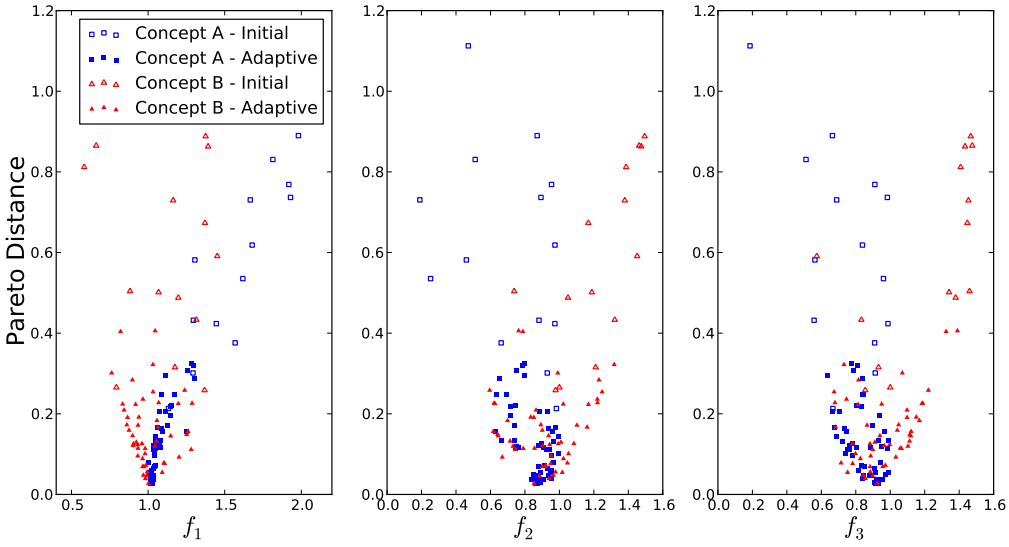


Figure 59: Pareto Distance Chart - Objectives

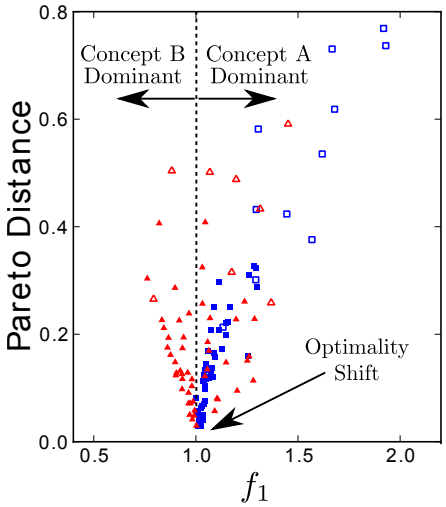


Figure 60: Pareto Distance Chart - Objective 1

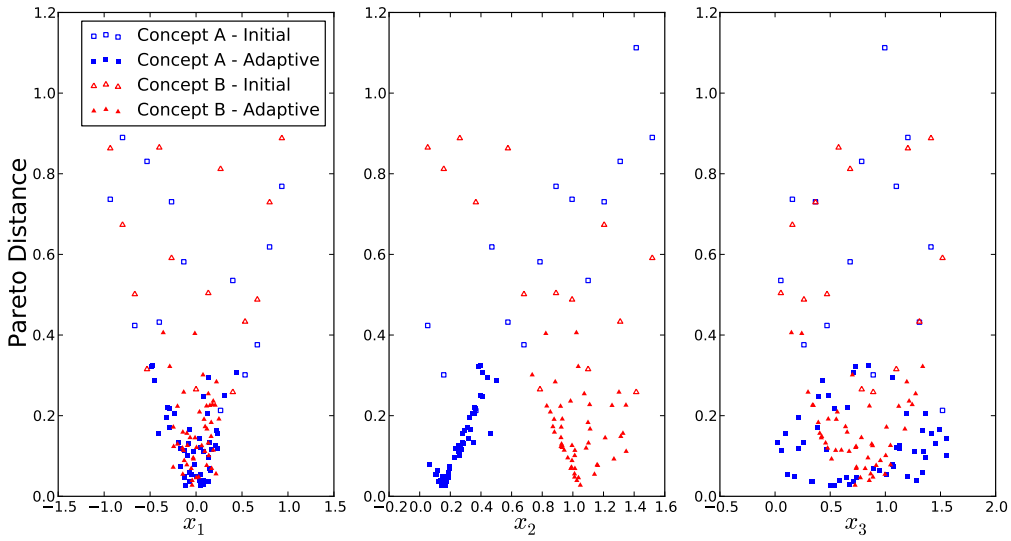


Figure 61: Pareto Distance Chart - Design Variables

61. A decision maker can easily identify the value of the design variable x_2 where Concept A becomes more preferred to Concept B.

Another conclusion to draw from Figure 61 is the independence of optimality shift to a particular design variable. Comparing x_2 and x_3, a designer can interpret the behavior as a derivative, or rate of change. In other words, the question of how sensitive concept optimality to the design variables can be answered by observing the slope of the data as it moves away from $PD = 0$. In this example, a small change in x_2 leads to a rapid departure from the Pareto intersection. This is contrasted with the design variable x_3 where designs remain close to the Pareto frontier intersection over a wider range of values (lower slope). The reader should take care to note that shallow slope does not indicate objective independence to a particular design variable. Rather, the *shift in optimality* is less influenced by changes in the concept design. It is important to remember that while Concept A and B both have a design variable called x_2, these are in fact independent. The simplicity of the problem and similar ranges enable the two sets of designs variables to be plotted together.

The algebraic problem is expanded to four objectives, and the experiment is repeated similar to the three objective case. Pareto distance is used again to visualize the frontier intersections in two-dimensional projections. Sampling results and corresponding Pareto Distance Charts are shown in Appendix C.2.

5.2.3.3 Experiment 3.2.2: Algebraic Sample Modification

The results obtained up to this point show promising performance for an efficient evaluation method aimed at locating the Pareto frontier intersections of competing concepts. The algebraic sample problem discussed above is now modified to investigate performance of the proposed method on concave Pareto frontiers. As mentioned in Chapter 2, a desirable quality of any concept evaluation approach is to avoid being fooled by deceptive functions. In other words, the proposed method should not rule out Pareto optimal designs even though they are in a concave region of Pareto optimality. Another quality of this example will allow exploration of the sampling behavior when the Pareto frontier intersection is a locus of points rather than single point in the objectives space. The three-objective algebraic sample is defined by the following equations.

$$f_1^{(k)}(x) = M^{(k)}(N^{(k)} + x_3)cos(x_1)cos(x_2 + \frac{\pi}{2}) + O^{(k)} \quad (44)$$

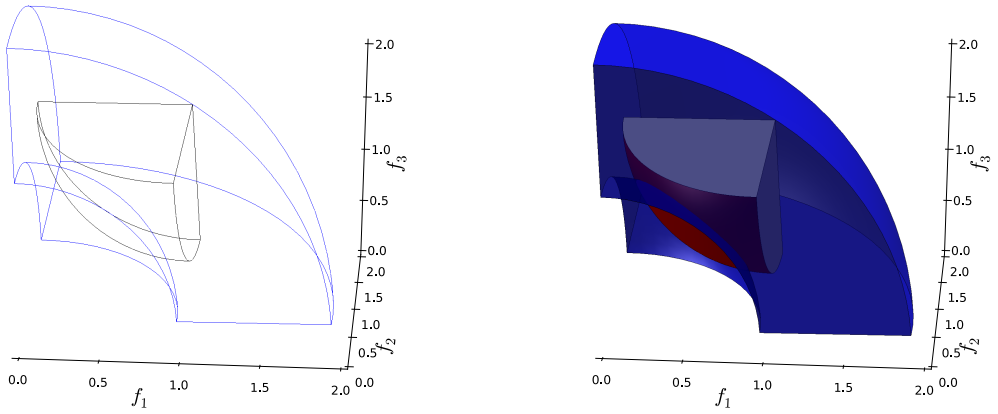
$$f_2^{(k)}(x) = M^{(k)}(N^{(k)} + x_3)sin(x_1)cos(x_2) + O^{(k)} \quad (45)$$

$$f_3^{(k)}(x) = M^{(k)}(N^{(k)} + x_3)sin(x_1) + O^{(k)} \quad (46)$$

$$\text{Subject to:} \qquad 0 < x_1 < \frac{\pi}{2}, \quad -\frac{\pi}{4} < x_2 < -\frac{\pi}{4}, \quad 0 < x_3 < 1,$$

where $[M^{(A)}, N^{(A)}, O^{(A)}] = [1, 1, 0]$ and $[M^{(B)}, N^{(B)}, O^{(B)}] = [-1, 0, 1]$.

The three-dimensional objective space is shown in Figure 62. The goal again is to minimize the three objectives. Concept A (blue solid) has a concave, spherical Pareto frontier that is intersected by Concept B (red solid). For this example, the



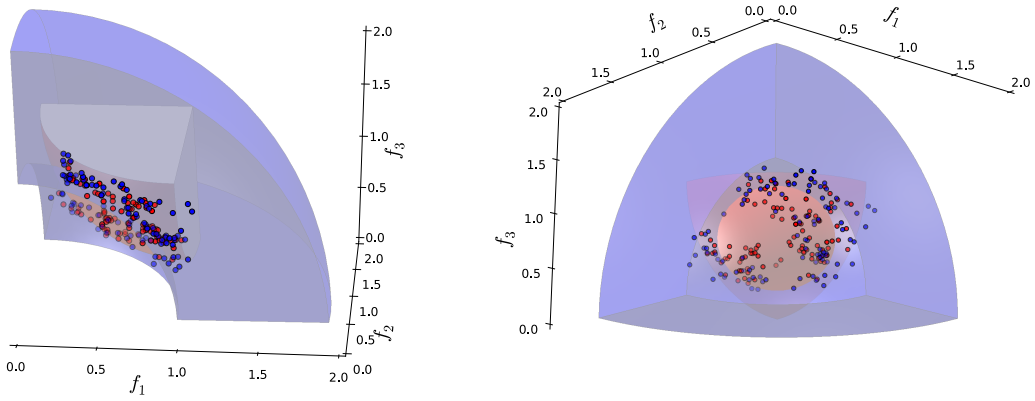
(a) Wireframe (b) Solid

Figure 62: Modified Algebraic Sample Problem

intersection is defined by a small circle of designs in the center of the concave Pareto frontier of Concept A.

The PFI iteration is performed again under the same assumptions as the first experiment. The iteration is run for 100 samples of both Concept A and B to provide a more dense set of points from which to draw conclusions about the more difficult problem. The sampling results are shown in Figure 63. Two projections again are shown for clarity. In Figure 63a, the samples can be seen to reside in the region of s-Pareto optimality (all objectives minimized). This view however, does not depict very accurately the circle of intersection between the two concepts. The perspective presented in Figure 63b is perpendicular to the plane formed by the circle of intersection and provides a clearer picture of the intersection between the two concepts.

A number of observations can be made from Figure 63. First, the evaluation method avoids suboptimal regions of both concepts. The intersection is not as cleanly defined by the samples as the test problem presented above, but some desirable qualities are nevertheless evident. A majority of the sample data appear to be scattered in an arc-shape near the top right portion of the circle of intersection. While a perfect sampling would place designs only on the intersection of Concept A and B, this



(a) Projection 1 (b) Projection 2

Figure 63: Sampling for Modified Algebraic Problem

scatter is beneficial from the perspective of decision making. Knowledge of the exact intersection alone is not as useful as designs *close to* the intersection. That is, in which direction is one concept preferred over another? Without design clustering, the evaluation method would be incapable of predicting which concept is dominant as decision maker preferences move away from the intersection. The scatter to the data can be explained by the exploitative property of the infill criterion. In minimizing the Normalized Pareto Distance (NPD) component to PIC, designs are sought that minimize distance to competing known designs. A sparse initial sample will lead to the situation where designs may be placed near existing Pareto optimal points, which may be far from the true intersection or s-Pareto optimality.

5.2.3.4 Experiment 3.3: Scalability

At each iteration, a global search on PIC must be performed. The total cost to set up and execute a PFI framework therefore, may represent a large amount of computational work relative to executing the concept models. When these models are sufficiently expensive and each function call is at a premium, designers are willing to incur this expense to achieve a more intelligent search. For rapidly executing models, PIC-based evaluation may be impractical, and required accuracy can be obtained

with dense Monte Carlo sampling or traditional multiobjective optimization. The purpose of this experiment to is gain an understanding of how expensive a model must be to warrant PFI-based evaluation.

Wall clock time to optimize PIC was tracked at every iteration for the two, three, and four objective samples. While this time is strongly dependent on computer hardware and optimization method, general trends can be obtained. The experiments were performed on an Intel Core^{TM}2 Duo CPU with 2.53 GHz and 1.96 GB RAM. A genetic algorithm was used to perform the global optimization with the following parameters: real variable encoding, tournament selection, elitism, population of 50, five generations, crossover rate of 0.9, and mutation rate of 0.02. The specific implementation of the genetic algorithm was obtained from the Python library PyEvolve [7]. In the author's experience, these heuristics yielded favorable optimization results for the algebraic sample but are often highly problem dependent [70].

The plot shown in Figure 64 shows variation in computation time as the iteration progresses as a function of total samples. The samples are also separated into two, three, and four objectives. An initial population was assumed to be five times the number of degrees of freedom (equal to number of objectives) and explains the staggered starts to each data set: 10 for two objectives, 15 for three objectives, 20 for four objectives. There is an increase in time required to perform the optimization as sample size increases. This is due to the increased expense of evaluating PIC, which is dependent on number of current Pareto points.

Another important observation is the spread of data (noise in the measurement) for three and four objectives compared to a more linear trend with two objectives. This can be explained by the method used to calculate PIC. The two objective case uses a closed form solution of Probability of Improvement found in [85]. Rather than perform numerical integration directly, a fast approximation is achieved using the

property of the cumulative normal distribution function given by Equation 47.

$$\Phi(x) = \frac{1}{2}\left[1 + \text{erf}\left(\frac{x}{\sqrt{2}}\right)\right] \tag{47}$$

where $\text{erf}(x)$ is the Gaussian error function given by Equation 48.

$$\text{erf}(x) = 1 - \frac{2}{\sqrt{\pi}} \int_x^\infty e^{-t^2} dt \tag{48}$$

The function described by Equation 48 is a sigmoid curve and easily approximated by table-lookup or Taylor series expansion leading to rapid evaluation of MPPI, a major component of PIC, in two-objectives. Problems of more objectives rely on Monte Carlo sampling followed by Pareto filtering to approximate MPPI. The number of Monte Carlo samples is then left up to the designer. For this experiment, 100 MCS points were used to approximate MPPI in three objectives and increased to 200 for four objectives. While the number of samples is small compared to traditional MCS experiments, it is generally desirable to use the minimum number that gives a close enough approximation. A multivariate random number generator from the Python library NumPy [8] was used to obtain the jointly distributed set of points with mean $\hat{y}$ and sigma s^2 from the Kriging predictor for each member of the GA population.

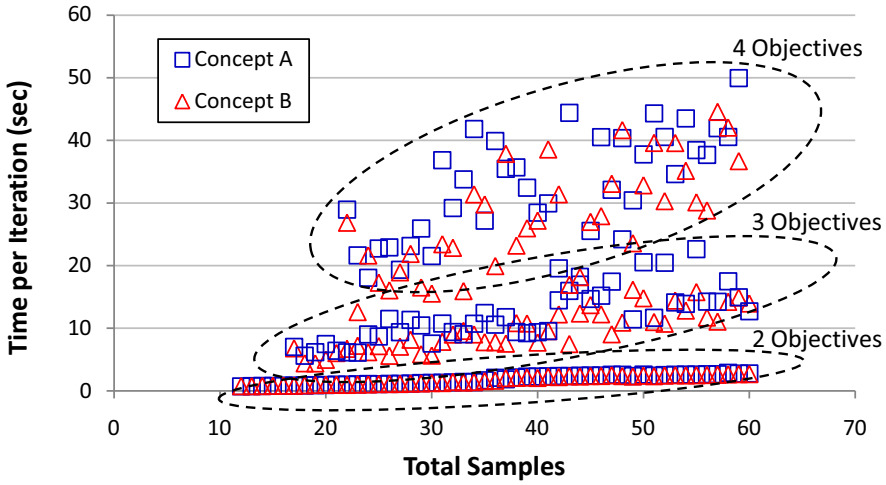


Figure 64: Time per Iteration as a Function of Total Samples

Cumulative time is shown in Figure 65. For two objectives, the cost of 60 PIC iterations took approximately 90 seconds while four objectives took almost 22 minutes for 40 iterations (ignoring concept model calls and initial Kriging training). This highlights a significant tradeoff in doing PFI-based evaluation. With the setup described above, a single function call of Concept A takes approximately 3×10^{-5} seconds. In the time it took to perform the overhead calculations for PFI-based evaluation, almost 1.5 million function calls could have been executed on each concept. For such a simple model, PIC sampling is unnecessary. For more expensive models with execution times on the order of seconds or minutes (not unreasonable for many conceptual design tools), the PFI overhead is less severe when function calls can be reduced significantly.

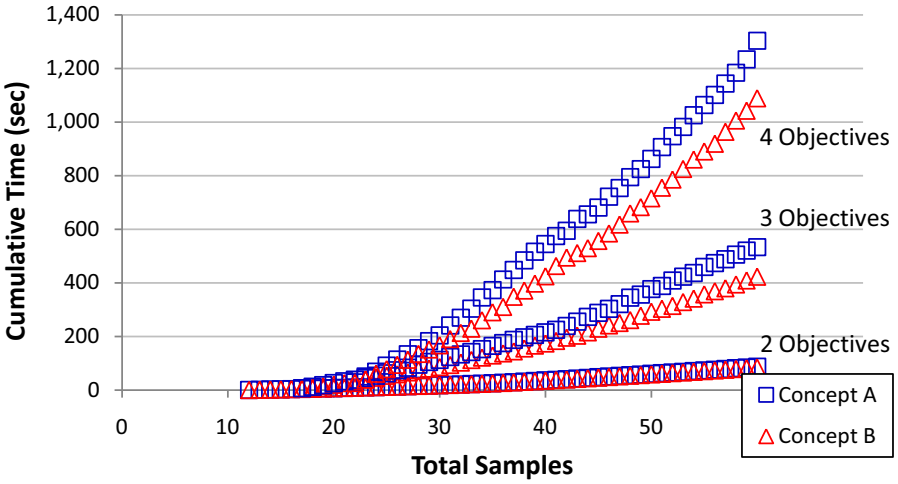


Figure 65: Sampling Cumulative Time

To assess the efficiency of PIC, the number of optimal points is plotted as a function of total samples in Figure 66. Ideally, two optimal points will be found for each iteration, one from each concept. This assumption is valid only for those problems where the two concepts make up the s-Pareto frontier. The formulation of PIC infill criterion means that perfect sampling would sample only points on the s-Pareto frontier. This is shown in the figure as dotted lines with a slope of one

(i.e. one adaptive sample augments the current Pareto set by one). Note that space-filling samples are ignored. Performance can then be assessed based on deviation of actual sampling from the ideal case. The sampling efficiency appears independent to number of objectives as deviation from ideal is similar for the three experiments. This deviation can be caused by two phenomena. First, the adaptive point is not optimal leading to identical number of optimal points on subsequent iterations. Second, a dominating design is found which leads to elimination of one or more prior designs. This situation is more severe if the iteration were terminated early, and true Pareto optimal performance had yet to be discovered.

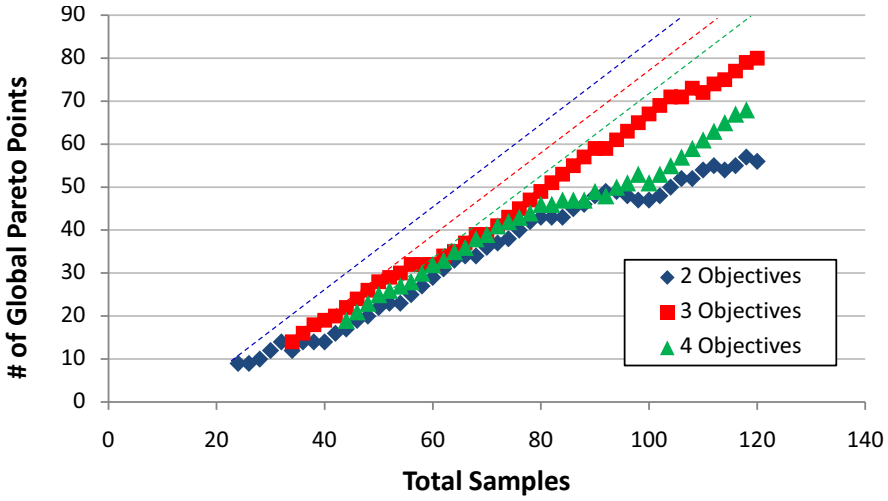


Figure 66: Sampling Efficiency

5.2.4 Experiment 4: Validating Assumptions

5.2.4.1 Experiment 4.1: Sampling Split

The dependence on the ratio of initial samples to adaptive samples is now investigated for the bi-objective problem. Maintaining a total budget of 60 total function calls, the experiment is repeated for the following initial/adaptive splits: 6/54, 8/52, 20/40, 30/30, 40/20, 50/10. The purpose of the experiment is to identify the maximum number of initial samples that avoids evaluating too many suboptimal designs. The sampling results are shown in Figure 67.

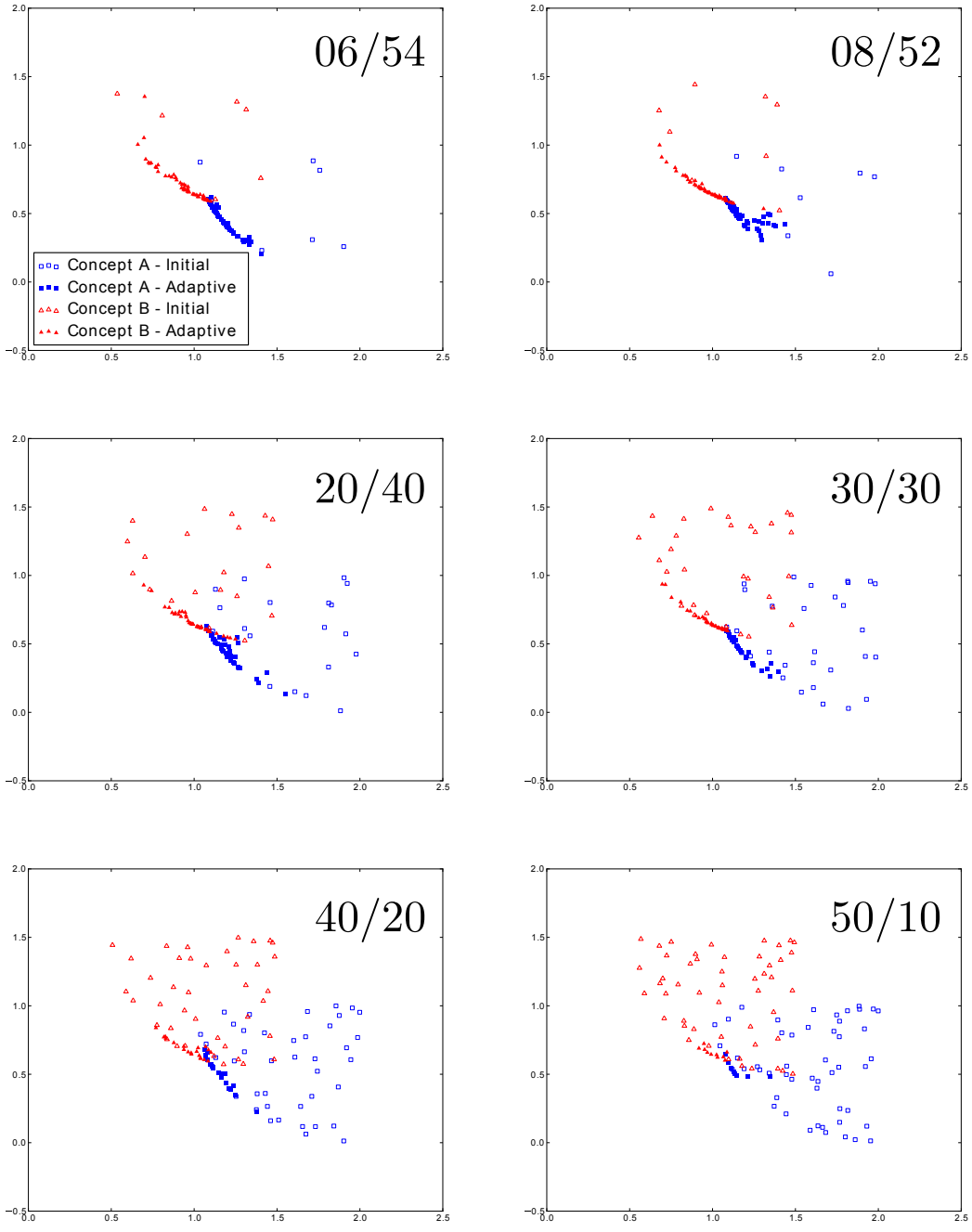
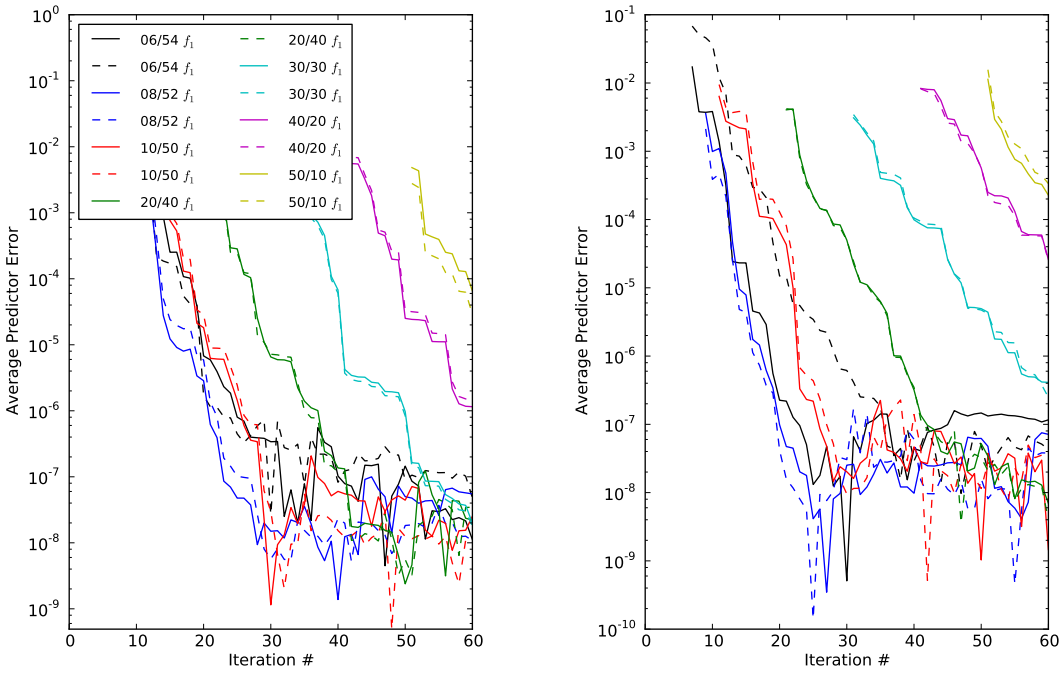


Figure 67: Sampling for Variation in Initial/Adaptive Ratio

As the ratio of initial to adaptive samples decrease (i.e. number of adaptive samples grows for a fixed number of total function calls), more design points are executed in the area where preferences shift from one concept to another. Even for the smallest initial sample, all adaptive samples appear to lie on the s-Pareto frontier. This is due in part to the simplicity of the algebraic sample and ease with which Kriging predicts the model even with a few training points. For the 50/10 split, the majority of model executions are merely space filling (bottom right corner of Figure 67. The top-left sampling scheme is more informative when selecting between Concept A and B because the region where the selection decision shifts is modeled very accurately. This is supported by the iteration history of model uncertainty at the intersection. Predictor error at the precise intersection (Region 1) versus iteration number is plotted Figure 68.



(a) Concept A

(b) Concept B

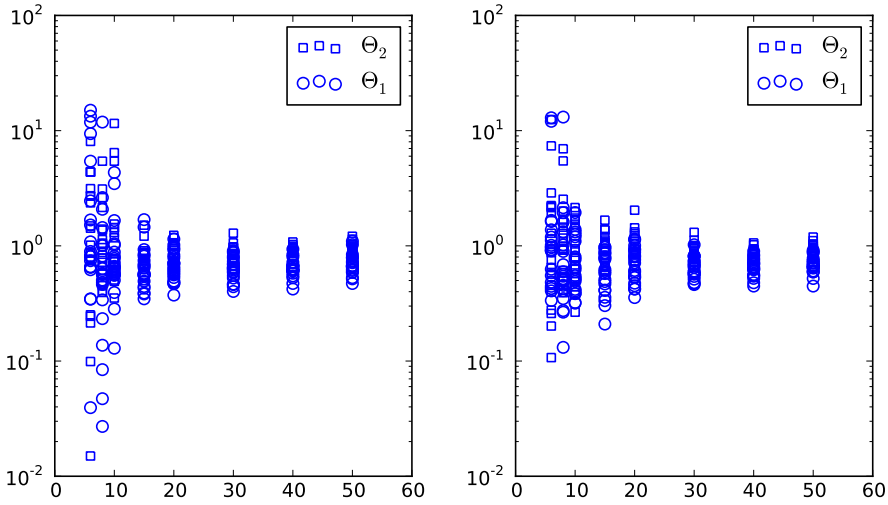
Figure 68: Adaptive and Initial Sample Tradeoff

From the above figure, it is apparent that the 50/10 split is the most inefficient, placing too many samples that do nothing to reduce uncertainty in areas designers truly care about. As adaptive samples are increased, there are diminishing returns. This is seen by the similar performance of the 06/54, 08/52, and 10/50 splits. In fact for this sample problem, 08/52 (blue line) outperforms the case with a lower ratio of 06/54 (black line). Initial samples cannot be reduced indefinitely as the space-filling sample must be used to train the first set of surrogate models. In general however, for the purposes of reducing uncertainty in areas where optimality shifts between concepts, fewer initial samples are preferred.

5.2.4.2 Experiment 4.2: Kriging Hyperparameter Tuning

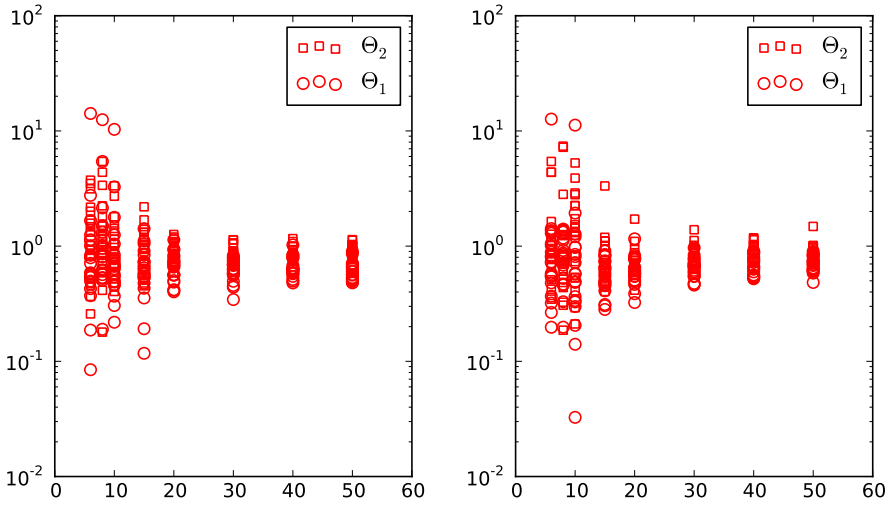
The purpose of this experiment is to investigate the sensitivity of the Kriging training assumption aimed at increasing sampling efficiency. In the above experiments, the tuning of the Kriging hyperparameters Θ_{x_1} and Θ_{x_2} (Equation 12) corresponding to the two degrees of freedom for each concept were assumed constant after each adaptive sample. This saved significant computational cost as only the correlation matrix R (Equation 13) was updated with each new sample. The log likelihood function given in Equation 14 was maximized once after the initial space-filling DOE was run.

This assumption can have important consequences where too few initial samples may lead to poor estimation of the Kriging hyperparameters. For that reason, a minimum number of space-filling designs must be run to properly characterize the design space. A space-filling DOE of increasing sample size was run for each concept and Kriging models trained on each set. The experiment was repeated 20 times to investigate training repeatability. Kriging hyperparameters were tracked as a function of total DOE samples and plotted in Figures 69 and 70 for Concepts A and B respectively.



(a) f_1 (b) f_2

Figure 69: Kriging Hyperparameter Trends - Concept A



(a) f_1 (b) f_2

Figure 70: Kriging Hyperparameter Trends - Concept B

As number of training data grows, the Kriging hyperparameters expectedly become more independent to number of total samples. Given n training points with n relatively large, sample $n+1$ probably will not provide new information with which to train the surrogate models. For small n however, a new sample may impact the behavior of the response greatly. Remember, the Θ's are measures of the sensitivity of the response to changes in a particular design variable. The purpose of the experiment is then to locate the point where Kriging hyperparameters begin to remain constant. Put in other terms, the goal is to locate the minimum number of samples for which anomalous results disappear.

For sample sizes of less than 10, the Θ's are highly sensitive to the initial population. The sparse representation of the design space leads to a large spread of converged hyperparameters. An iteration based on single training could lead to underestimation of uncertainty, so more initial samples are required for the assumption to be valid. For the algebraic sample problem, Kriging hyperparameters begin to "settle in" at constant values around 15-20 samples. There is still some spread to the data however caused by the randomness in the Latin Hypercube DOE. A minimum population of 15 to 20 initial samples is an increase over the $5k$ assumption (initial population is five times the number of design variables k) made in experiments 1-3. However, the simplicity of algebraic sample and ability of the Kriging to predict the response allowed for still accurate results. Based on the results of Experiment 4.2, further designs studies will assume a "10k" rule. This general trend is corroborated by Jones et al. who did much work to popularize Bayesian adaptive sampling for expensive model optimization [84].

For models that are inherently difficult to fit with a surrogate or expensive enough that the $10k$ rule is still prohibitively time consuming, an alternative can be used which involves training at every n^{th} iteration. In other words, after every n adaptive samples, the hyperparameters for the Kriging models are found through likelihood

optimization. This allows for time saving with each iteration yet not constrained by the fit of the initial surrogates. Selection of n is a heuristic parameter and depends on the type of problem, number or objectives/design variables, cost per function call, and model complexity.

5.3 Truss Design

The concept evaluation methodology is now demonstrated on the common engineering problem of truss design. The implementation hypotheses will be tested on this less idealized problem whose designs parameters and objectives have physical meaning. Scalability and robustness will be tested through increased number of competing concepts (unexplored with the algebraic sample) whose relationship between one another is defined by physics.

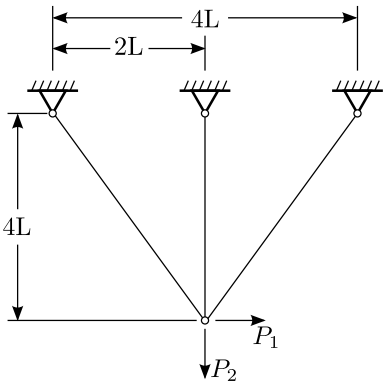
5.3.1 Problem Description

Six truss concepts are presented in Figure 71 with geometric properties governed by length L. Each truss has a single free node under an applied load. The designer would like to minimize total structural weight (or volume) of the truss as well as displacement of the free node with applied loads. The designer has available the cross-sectional areas of each bar as independent design variables. For this example, L is 30 inches, Young's Modulus E is $30\text{x}10^6$ psi, and the applied loads P_1 and P_2 are equal to 30,000 lbs. The cross sectional area of each bar is constrained to be between 1 and 4 inches.

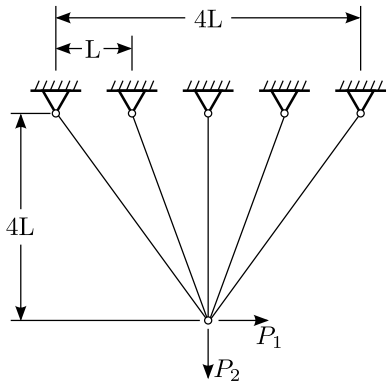
5.3.1.1 Assumptions and Modeling

In modeling the truss concepts, truss equations developed in [97] are used assuming linearly-elastic elements with constant cross-sectional area that obey Hooke's law:

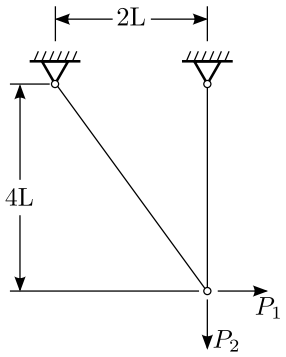
$$\sigma_x = E\epsilon_x \tag{49}$$



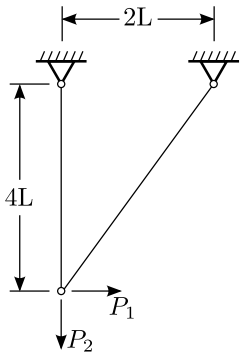
Concept A



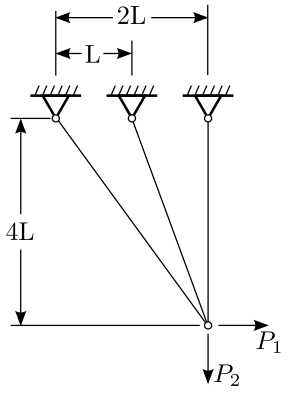
Concept B



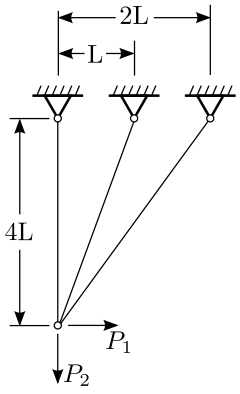
Concept C



Concept D



Concept E



Concept F

Figure 71: Six Truss Concepts

where σ_x is the stress along the x coordinate, E is Young's modulus of elasticity and ϵ is the strain defined by Equation 50.

$$\epsilon_x = \frac{d\hat{u}}{d\hat{x}} \tag{50}$$

where $\hat{u}$ is the axial displacement along the $\hat{x}$ direction. The truss elements cannot sustain shear forces and pinned ends constrain displacement (rotation allowed). Displacement in the truss members can then be found by solving Equation 51 for d.

$$F = \mathbf{K}d \tag{51}$$

where F is the global nodal force vector, $\mathbf{K}$ is the global stiffness matrix, and d is a vector of nodal displacements. For this simple problem, execution of a single design is instantaneous. As number of bars grows for more complex trusses, solving Equation 51 can become computationally expensive as the global stiffness matrix must be inverted to find displacements. Computational time can also be increased by moving away from simple linear theory and towards high fidelity finite element analysis. The truss solver was implemented by the author in OpenMDAO and source code provided in Appendix B.2.

The space of alternatives is shown by Figure 72. To generate the figure, a Monte Carlo Simulation was performed for 200 samples of each concept. The designs variables were uniformly distributed in the range $x_i = [1, 4]$.

The six concepts spanning a wide range of performance along the two objectives can be seen as different colors in the figure. Before specifying preferences, multiple concepts are dominant and equally acceptable, Pareto optimal alternatives. Concept B, as the only five-member truss, is the most stiff yet has the highest volume. As preliminary visual inspection of the space indicates, the s-Pareto frontier is made up of at least Concept B, A, C while Concepts E and F are dominated. From the random sampling, it is still unclear whether Concept D is dominant. While Monte Carlo Simulation was used merely to provide an initial visualization of the space,

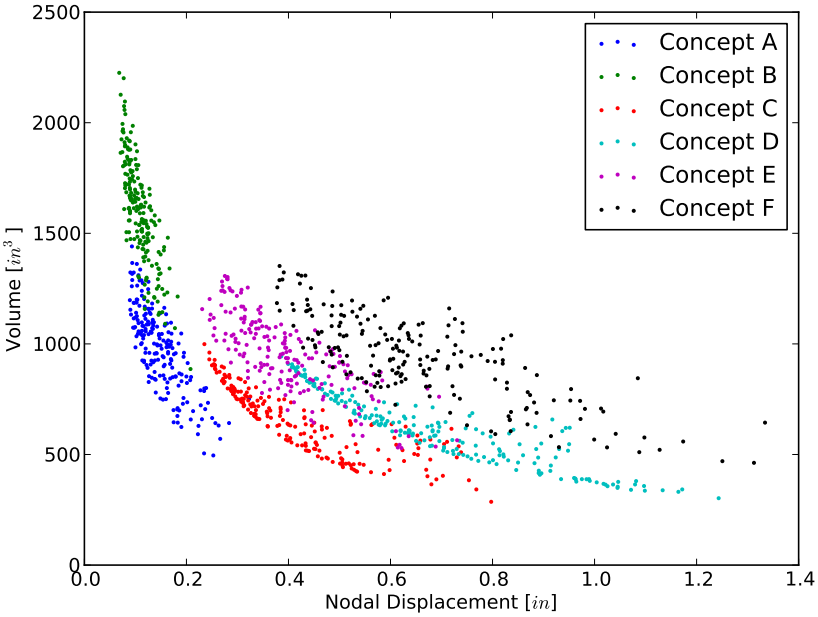


Figure 72: Objective Space for Six Truss Concepts

for expensive analyses this may be infeasible. As number of design variables and objectives grow, a blind search such as MCS yields even more sparse sampling along the Pareto frontier.

5.3.2 Results

The design spaces of all six concepts were initially seeded using a Latin Hypercube DOE with number of samples corresponding to ten times the number of degrees of freedom (the $10k$ rule from prior experiments). The space-filling sampling is shown in Figure 73.

The concepts were adaptively sampled for 50 iterations using the PFI-based approach. The iterative samples are shown in Figure 74. Upon inspection, the s-Pareto frontier appears to be sampled heavily compared to dominated designs. In addition to locating the optimal points, there is clustering of designs in areas where optimality shifts between concepts. This can be seen by examining the intersections more closely. Figure 75 shows two zoomed-in regions from the total objective space of the

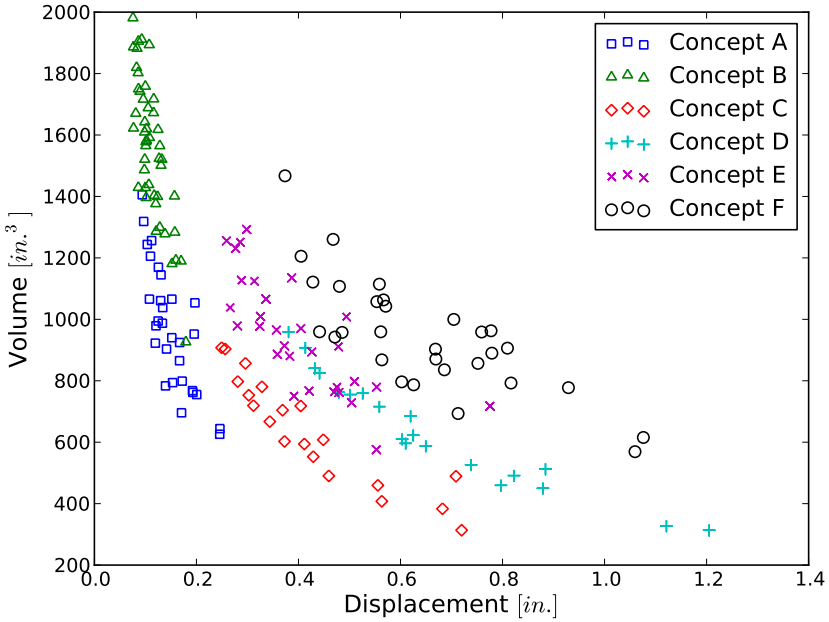


Figure 73: Space-Filling Sample for Truss Design Problem

six concepts depicted in Figure 74. The relationship between Concept A and B can be seen quite accurately in Figure 75a, and in fact, many samples were placed off the s-Pareto frontier in an attempt to be closer to their Pareto intersection. As distance from this point increases, sampling becomes more sparse for both concepts. This desirable quality allows better characterization of regions where optimality shifts from A to B while sacrificing accuracy where concept dominance is well understood. In addition to heavily sampling Concept A near the intersection with B, six adaptive samples were placed in the region where preference shifts from A to C. This dual-clustering can be expected for a multi-concept evaluation in two objectives where samples are placed towards the two extremes of a concept's optimal set (for those concepts on the interior of the s-Pareto frontier). The extremes correspond to a shift in optimality between concepts and can also be observed in Concept C samples (red diamonds). It is important to note that the Pareto designs are ignored in between these extreme regions for Concept A and C. The reason for this is not because these

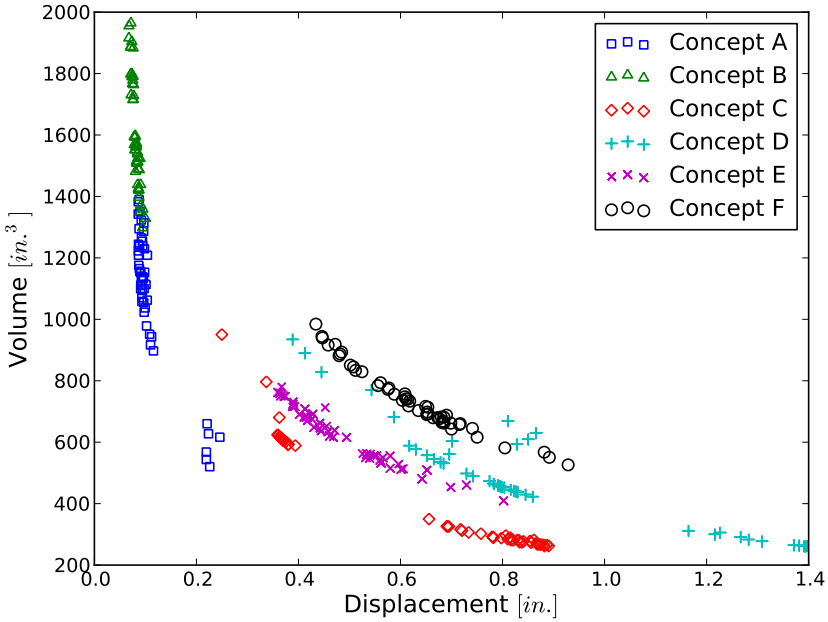


Figure 74: Adaptive Samples for Truss Design Problem

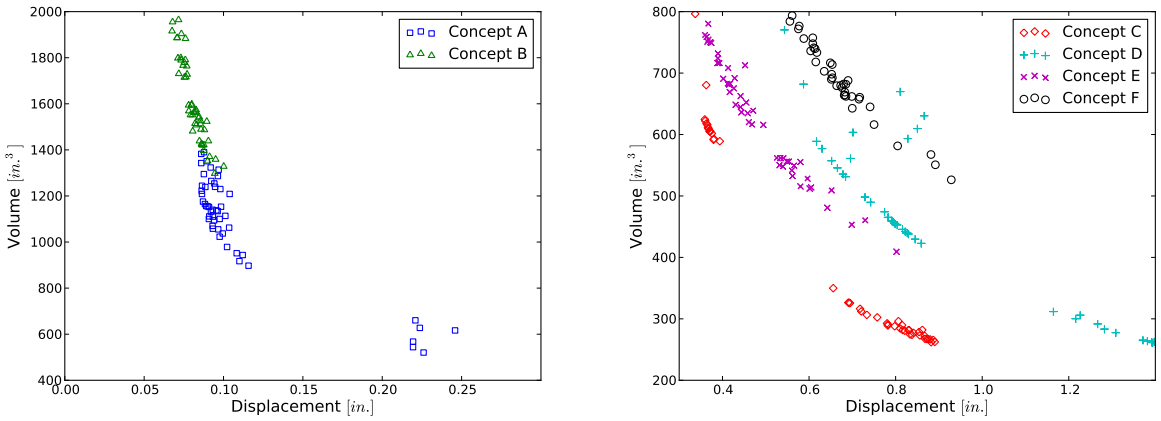
areas are not important, but concept dominance is already known at these locations. Another key observation is the size of the clusters at each extreme. For instance, many more samples of Concept A were placed near B than C. This is explained from the definition of NPD, which attempts to strictly minimize the distance from other known Pareto designs. Since A and B have a true intersection of Pareto frontiers, the distance between designs is much smaller. The s-Pareto frontier is discontinuous between A and C which creates a larger distance and biases designs towards the continuous Pareto intersection.

Figure 75b shows the four concepts (C, D, E, and F) representing the lower volume, higher displacement alternatives. A number of conclusions can be drawn from this figure. Concept D, while only a small portion of its own frontier is also on the s-Pareto, shows a small clustering of designs where optimality shifts from C to D (teal crosses). The remaining D samples are placed in areas that are *most likely* to be s-Pareto optimal. The same can be said of Concept E and F samples. While the

samples were placed on or near the Pareto frontiers of individual concepts, these concepts are entirely dominated by other truss alternatives.

One final comment should be made about a potential drawback when using PIC as an infill criterion. A phenomenon only partially observed in the truss design example has the potential to place designs well off even the individual Pareto frontier of certain concepts (Concept D in this example). This can be explained by the probability of improving (MPPI component to PIC) dropping below machine precision. In other words, the concept has been modeled so well that the uncertainty in this area is very low. As a result, the tails of the joint probability distribution on new candidate points are very small in regions of Pareto optimality. While there is still a finite probability of improving based on the Gaussian posterior (a design *may* exist that improves upon all other designs), it is just very small. When MPPI goes to zero, this leaves only the NPD contribution to PIC which leads to what appears to be random sampling or clustering in suboptimal regions. This is highly undesirable and has the effect of drastically reducing sampling efficiency. However, in practical implementations of PFI-based evaluation with complex models, it is expected that this phenomenon will not manifest until many samples have been run in the analysis. Until this point, it is assumed surrogate errors will remain sufficiently large to avoid these numerical precision issues and always place samples in desirable (or suspected desirable) locations.

In Figure 76, both initial and adaptive samples are filtered for each concept to obtain only those designs that are on the Pareto frontier of their respective concepts. The results are filtered further to obtain those designs that reside on the s-Pareto frontier (shown in Figure 77). Examination of the figures highlights the tendency of PFI-based evaluation to ignore areas of the design space that are known to be dominant. This is especially true with Concept C where the Pareto frontier is most accurate at the extremes and jagged on the interior. It is important to note that



(a) Region 1 (b) Region 2

Figure 75: Zoomed Adaptive Sampling for Truss Design Problem

precise definition of Pareto optimality is less important here because of inherent uncertainty of the problem and modeling assumptions made at this stage in conceptual design. Clusters of designs are more important to unearth trends in design variables that cause a shift in dominance from one concept to another.

Computation time for each iteration was tracked for the truss design and presented along with the bi-objective algebraic sample in Figure 78. Again, initial surrogate training and filtering are assumed negligible. Figure 78a shows time as a function of total samples. Each concept was evaluated at 50 adaptive samples, but had an initial population corresponding to the number of design variables. The starting number of total samples is thus staggered according to this difference in initial sampling between concepts. Figure 78b shows the same data but as a function of iteration number which is uniform between truss concepts.

While not expected to be exactly that of the algebraic problem, the time per iteration is similar in magnitude for the truss evaluations: on the order of a few seconds. The time increase for the truss evaluation can be attributed to the increased number of Pareto points derived from six concepts rather than just two. Another factor that affects iteration time for a particular concept is the number of design

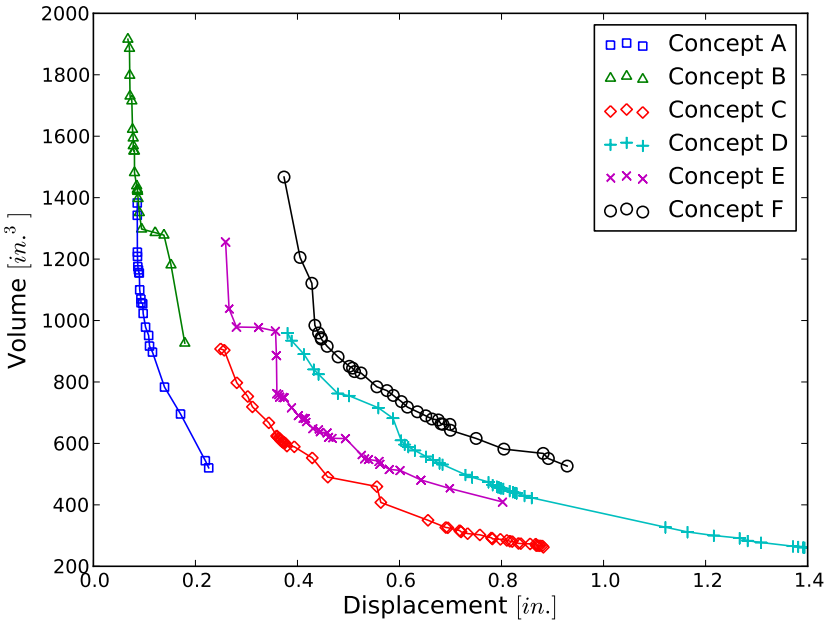


Figure 76: Individual Pareto Frontiers for Six Truss Concepts

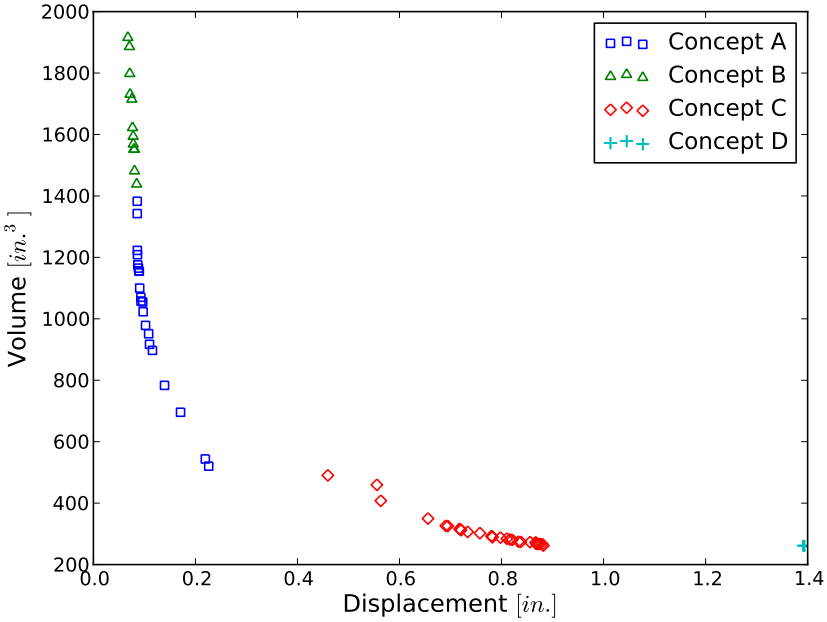
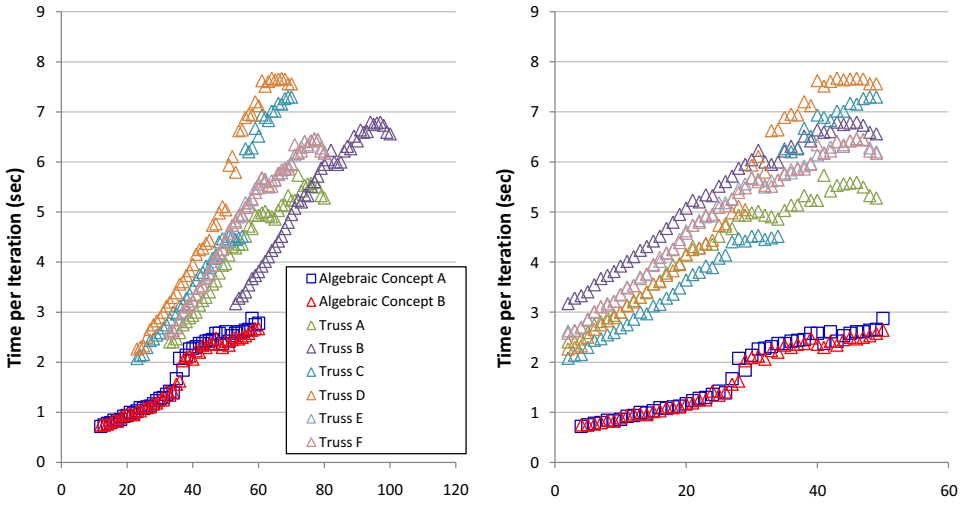


Figure 77: Optimal Designs for Six Truss Concepts

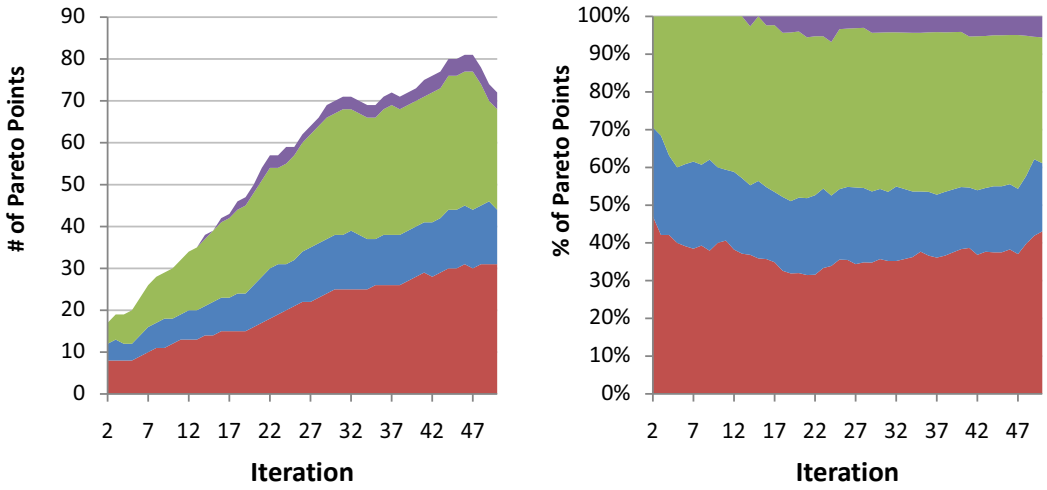


(a) Total Samples (b) Iteration

Figure 78: Time per Iteration for Canonical Problems

variables. For the early iterations, this trend is evident in Figure 78b where the concept with the five design variables (dark purple triangles) is the most expensive evaluation compared to other alternatives. However, as the iteration approaches around 30 samples, a different behavior is observed where evaluations of Concept C and D (teal and orange triangles) dominate computation time. In other words, the slopes of these two line lines rapidly increase compared with other concepts. To explain this phenomenon, another factor affecting computation time, number of known optimal points, must be investigated further.

Figure 79 shows the sampling efficiency (iteration number vs. number of optimal points) of the truss design evaluation along with contributions to the total s-Pareto frontier from each concept. Both absolute (left) and percentage contribution (right) are presented. As the iteration progresses, Concept C and D have gradually more of a contribution to the set of optimal designs. This contributes to the increased time per iteration depicted in Figure 78. In fact, the number of Pareto points from each concept is the only thing that varies during the iteration and also explains the increase in computational time for the truss example compared with the algebraic



(a) Absolute (b) Percentage

Figure 79: Contribution to s-Pareto Frontier

problem.

5.4 Conclusions

Some general conclusions on both the algebraic sample and truss design problems are presented here with a revisiting of the hypotheses presented in the beginning of this chapter. The inexpensive function calls of relatively simple models allowed more focus on experimentation, trade studies, bug fixes, and further development of the methodology. When executing the PFI-based evaluation, the particular analysis codes can be thought of as black boxes with the caveat that they are sufficiently modelable with Kriging. For that reason, these canonical problems present a strong indication of method performance for the more complex problem of UHB engine design whose competing concepts are modeled with deterministic and continuous computer codes.

5.4.1 Revisiting the Hypotheses

Results from the above experiments are summarized with regard to the hypotheses presented in Section 5.1.1. The results from Experiment 1 showed the method's ability to concentrate computational effort in regions where Pareto optimality shifts between

concepts. This supports Hypothesis 3.1 as the PIC infill criterion balances the search for Pareto optimality among all concepts with similar performance between designs. Plots of adaptive samples show a clustering of points near the Pareto intersection of competing concepts both for algebraic sample and truss design. For the case where the intersection can be located analytically, predictor error was reduced relative to less desirable regions of the design space.

While all investigated adaptive sampling methods (MOPI, MPPI, and PIC) can reduce predictor error near the intersection, PIC is the only sampling criterion to focus specifically on locations where optimality shifts between concepts. Experiments on the sample problem support Hypothesis 3.2 where the PFI-based approach reduces predictor error at the intersection with the fewest function calls. However, this ignores the additional computational cost of the PFI evaluation framework and optimization of the infill criterion at every iteration. If concept models are inexpensive like in the case of truss design, a more practical solution is to use a zero-overhead method such as Monte Carlo or evolutionary optimization where many more cases can be executed.

For the canonical problems, the method is generally robust to relationships between concepts regardless of existence of a true Pareto intersection. For those cases where an intersection does not exist, intuitive behavior is exhibited. Evaluation of completely dominated concepts reduces to search for designs that are similar to the optimal concept's designs. For analyses with more than two optimal concepts, an important phenomenon was identified where samples may become biased towards a single optimality shift. This was demonstrated for the truss design example and may be more significant on other problems. In these situations, the author recommends removing one or more of the concepts from the analysis once the intersection has been modeled to the desired accuracy. This will eliminate ambiguity over which optimality shift is of interest.

The method is just as capable at locating optimality shift in three, four, or n

dimensions as in two. The difficulty remains visualizing the results and identifying where each concept becomes dominant. The author proposes a new way of visualizing Pareto optimality through a Pareto Distance metric. When PD is plotted as a function of the designs variables and objectives, the point at which optimality shifts between concepts can be more easily deduced. While overhead costs of a PFI-based evaluation framework increases with number of design variables, objectives, and concepts, this cost will be nominal compared to the cost of executing a sufficiently expensive analysis code.

The final experiment investigated two major assumptions of PIC sampling: initial/adaptive sampling split and Kriging hyperparameter tuning strategy. The results and lessons learned from the experiments serve as ground rules for large-scale problems.

CHAPTER VI

IMPLEMENTATION ON THE UHB DESIGN PROBLEM

6.1 Step 1: Define the Problem

The next generation, single-aisle transport with capacity around 150 passengers will make up a large portion of new aircraft entering service in the next five to ten years. Boeing and Airbus are expected to compete for replacement of their 737 or A320 aircraft in the 2015 timeframe. Demands on these aircraft will be even greater as designers must reduce noise and emissions to minimize the environmental impact of civil aviation at the same time improving performance and fuel efficiency. The specific problem to be addressed is engine selection for the next-generation single-aisle transport. Ultimately, designers would like to investigate if the efficiency benefit from UHB engine technology is enough to counteract the weight and drag penalty associated with under-wing installation on this class of commercial aircraft.

6.1.1 Objectives

The quantifiable metrics that encompass all the performance, economic, and environmental goals have been identified by NASA's Fundamental Aeronautics Program and listed in Table 11. These aggressive goals are all competing, and achieving a minimum value of each is not possible. In fact, the single objective solution along each metric may possess vastly different engineering and design characteristics. The most preferred, profitable aircraft with be a balance between all four objectives.

A more detailed description of each metric is now provided.

Table 11: Metrics for Engine Architecture Study

	Objective	**Nomenclature**	**Units**
1	Block Fuel	W_{fuel}	lb
2	Cumulative Noise Margin	$Cum\,Noise$	EPNdB
3	Ramp Weight	W_{ramp}	lb
4	Exhaust Emissions	$LTO\,NO_x$	$\frac{g}{kN}$

6.1.1.1 Block Fuel

Total fuel used during the mission, measured in pounds, represents an economic and performance metric that aircraft and engine designers would like to minimize. Fuel is the largest component to airline operating costs and impacts directly the economic viability of new commercial planes [114]. Required fuel also has range and payload implications as well as direct correlation to carbon dioxide emissions, an important greenhouse gas.

6.1.1.2 Ramp Weight

Ramp weight is defined as the maximum permissible aircraft gross takeoff weight, measured in pounds. This metric is often used as a surrogate for vehicle acquisition cost when detailed cost models are not available. Combined with block fuel as a measure of operating cost, ramp weight offers designers an early prediction of total aircraft life-cycle cost.

6.1.1.3 LTO NO_x

The exhaust emission indices ($EINO_x$) are measured in grams of nitrous oxides (NO_x) emitted from the engine per kilogram of fuel burned. Nitrous oxides consist of both nitrous oxide (NO) and nitrogen dioxide (NO_2) and are formed in a turbine engine when nitrogen in the air oxidizes from the extreme heat of the combustor. NO_x poses a health hazard to plant and animal life both on the ground and upper atmosphere where ozone depletion is a concern. The precise aircraft certification metric is of nitrous oxides emitted during the landing and takeoff (LTO) portions of a typical

mission. This factors in all operations near airports and accounts for those phases of flight when NO_x emissions are greatest, maximum thrust takeoff condition when combustor reaction temperatures are highest. $EINO_x$ is adjusted based on rated thrust to provide a certification measurement in the grams of NO_x per kiloNewton thrust. Two levels of stringency are imposed by the FAA (FAR 36) based on engine manufacture date and correlated to overall engine pressure ratio (OPR) [1].

- Before December 31, 1999: LTO $NO_x \leq (40 + 2OPR)$
- After December 31,1999 - LTO $NO_x \leq (32 + 1.6OPR)$

6.1.1.4 Cumulative Noise

The noise metric is defined as the summation of aircraft noise observed at three measuring stations near a runway.

Flyover - measured directly under the aircraft 6500 from the start of take-off ground roll

Sideline - measured 450 meters perpendicular to the runway centerline at the lateral point where aircraft noise is greatest

Sideline - measured 450 meters perpendicular to the runway centerline at the lateral point where aircraft noise is greatest

Approach - measured directly under the aircraft when at a distance of 2000 meters from the runway

These microphone locations are shown in Figure 80.

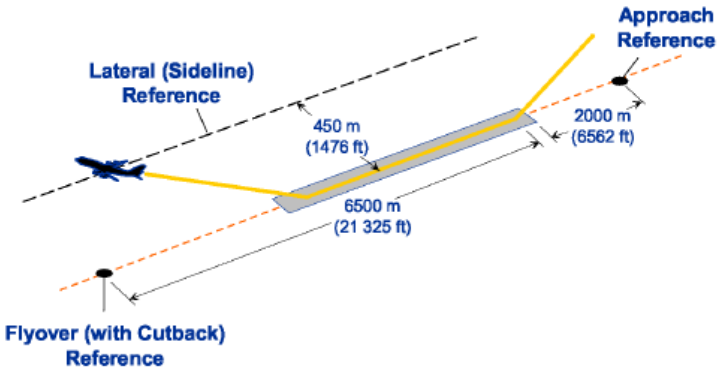


Figure 80: Noise Certification Points [23]

The noise metric defined by the FAA is called Effective Perceived Noise Level (EPNL) with units of EPNdB as listed in Federal Aviation Regulations Part 36 (FAR 36). EPNL depends on perceived noise level measured by microphones and is corrected for spectral irregularities, tone, and duration. The noise metric will be assessed relative to current Stage 4 levels and so represents a margin below these stringencies.

6.1.2 Mission Requirements

In addition to the objectives to be minimized, there exists a set of performance requirements that must be met in order to perform the mission. These requirements are posed as constraints on the multiobjective design problem. There are seven total requirements placed on the aircraft: takeoff and landing field lengths, approach velocity, climb rate at top-of-climb condition (service ceiling), excess fuel, and excess thrust for second-segment (SS) climb and missed approach (mapp). These are summarized in Table 12.

Table 13: UHB Engine Concept Design Variables

	Objective	**Nomenclature**	**Units**
1	Wing Area	S_W	ft^2
2	Sea-level Static Thrust	$F_{N,SLS}$	lb
3	Fan Pressure Ratio	FPR	–
4	Overall Pressure Ratio	OPR	–
5	Compressor Work Split	PR_{split}	–

Table 12: Constraints for the Next Generation Single-Aisle Transport Aircraft

	Description	**Nomenclature**	**Limit**	**Units**
1	Takeoff Field Length	D_{TO}	$\leq$ 7,000	ft
2	Landing Field Length	D_L	$\leq$ 7,000	ft
3	Approach Velocity	V_{app}	$\leq$ 150	kts
4	TOC Rate of Climb	$\dot{h}_{TOC}$	$\geq$ 300	ft/min
5	Excess Fuel Weight	W_{ef}	$\geq$ 0	lb
6	SS Excess Thrust	$F_{N,SS}$	$\geq$ 0	lb
7	mapp Excess Thrust	$F_{N,mapp}$	$\geq$ 0	lb

6.2 Step 2: Generate Concepts

The research will compare three concepts as potential solutions to the engine design problem. The first concept is a baseline technology engine with direct drive and fixed geometry nozzle. The competing concepts will be a geared driven fan architecture with and without a variable area bypass nozzle. The three concepts will be evaluated based on the above objectives.

6.2.1 Designs Variables

Through collaboration with NASA engineers at Glenn Research Center as well as benchmarking previous UHB studies, a set of key design variables were selected. They were chosen as the most important engine and airplane sizing parameters and expected to have the largest impact on overall system performance. The design variables are summarized in Table 13.

While the models are in fact independent, the three competing concepts share this common set of design parameters. Note that the PFI-based evaluation methodology

in general makes no assumptions about degrees of freedom, where design spaces may be identical, overlapping, or entirely disparate among concepts. Because the engine architectures are similar for this problem, they share similar design spaces.

Wing area is a classic aircraft sizing parameter and is considered an aircraft-only variable largely independent of engine design. Sea-level static thrust on the other hand is one of the most important engine sizing parameters and serves to scale the engine airflow to meet thrust requirements during takeoff. Fan pressure ratio has important implications on engine design and influences geometry, thermodynamic performance, and fuel efficiency. The last two variables (OPR and PR_{split}) are included to completely describe the work done by each compressor. PR_{split} is the division of work between the high pressure compressor (HPC) and booster (LPC). Specifying FPR, OPR and work split uniquely defines the pressure increase across the fan, LPC, and HPC. These in turn have a direct correlation with combustor inlet temperature which has strong influence over thermal efficiency as well as NO_x emissions and fuel burn. From an engine design perspective, variation in these parameters can be viewed as a change of core technology (high pressure spool). For that reason, OPR and PR_{split} are treated as discrete alternatives representing major design decisions within the engine. It can be argued that these discrete solutions are merely different concepts. For consistency, the discrete variables with floating point inputs are still considered "design variables" whereas fundamentally different models are concepts. Design problems with mixed continuous and discrete variables are common to many concept evaluation tasks in early design and further reinforces the need for an efficient and robust methodology.

Ranges of acceptable values for each design variable adopted from [23] and [66] are presented below. The continuous variable ranges are maintained wide enough to provide thorough coverage of potential designs. The discrete values are assumed to

be upper and lower technology limits.

$$1120 \leq S_W \leq 2240, \quad 20800 \leq F_{N,SLS} \leq 44800, \quad \begin{pmatrix} 1.3 \leq FPR_{geared} \leq 1.6 \\ 1.4 \leq FPR_{direct-drive} \leq 1.7 \end{pmatrix},$$

$$OPR = \begin{Bmatrix} 42 \\ 32 \end{Bmatrix}, \quad PR_{split} = \begin{Bmatrix} High \\ Low \end{Bmatrix}$$

With the exception of fan pressure ratio, all concepts have identical ranges on the design variables. The lower FPR's for the gear driven fans reflect the ability to increase fan diameter at the same time alleviating the low pressure spool shaft-speed mismatch. The work split variable is binary taking on values depending on the overall pressure ratio, also binary. A low work split indicates the booster does less compression (more compression in the HPC) relative to high work engine. For OPR of 42 the low/high work split is 42%/29% where percentage indicates how much of the OPR is done by the HPC. For OPR of 32, the low/high work split is 42%/31%.

6.2.2 Modeling

Propulsion and airframe design are highly coupled disciplines that require multiple tools for quantitative analysis. The resulting "meta-analysis" tool is a group of six NASA-developed analysis codes and correlations linked together to produce an aircraft/engine combination optimized for a set of requirements. These are listed in Table 14. The meta-analysis was first created for the studies presented in [66] and [23] and later adapted to a more object-oriented implementation for OpenMDAO by engineers in NASA Glenn's Multidisciplinary Design Analysis and Optimization (MDAO) Branch. Each modeling component is discussed in more detail in the following sections.

Table 14: Modeling Tools for UHB Engine Analysis

Discipline	**Analysis Code Name**
Aircraft Sizing	FLOPS
Aircraft Weights	PDCYL
Emissions	UEET Project Correlations
Noise	ANOPP
Thermodynamics	NPSS
Aeromechanical Design/Flowpath	WATE

6.2.2.1 Aircraft Sizing and Weights

The analysis code used to perform aircraft sizing and synthesis is Flight Optimization System (FLOPS) [104]. FLOPS is a multidisciplinary, FORTRAN-based mission analysis tool developed by NASA Langley Research Center in the 1980's. To improve upon some of the empirical weight estimation methods used in FLOPS, a more analytical tool for fuselage and wing weight was used called PDCYL [28]. Although the research focused on evaluation of propulsion system alternatives, the metrics forced a system level analysis where the aircraft is sized as engine design parameters change.

The baseline aircraft model was developed for the Boeing 737-800 using publicly available data of geometry, performance, and emissions. A VSP representation is shown in Figure 81. Many of the baseline weights were obtained from a document released by Boeing for airport planning [11]. These weights are summarized in Table 15.

Table 15: Reference Weights for the Baseline 737-800 Aircraft

Weight	**Reference Weight(lb)**
Maximum Ramp Weight	174,700
Maximum Landing Weight	146,300
Operating Empty Weight	91,300
Payload Weight	32,400
Fuel Weight	46,063
Calibrated Mission Weight	170,123

Using these weights, a baseline mission range of 3060 nautical miles (nmi) was

Figure 81: Planform of Advanced Single-Aisle Transport

obtained from the payload range chart from [11]. This is shown as the red dot in Figure 82.

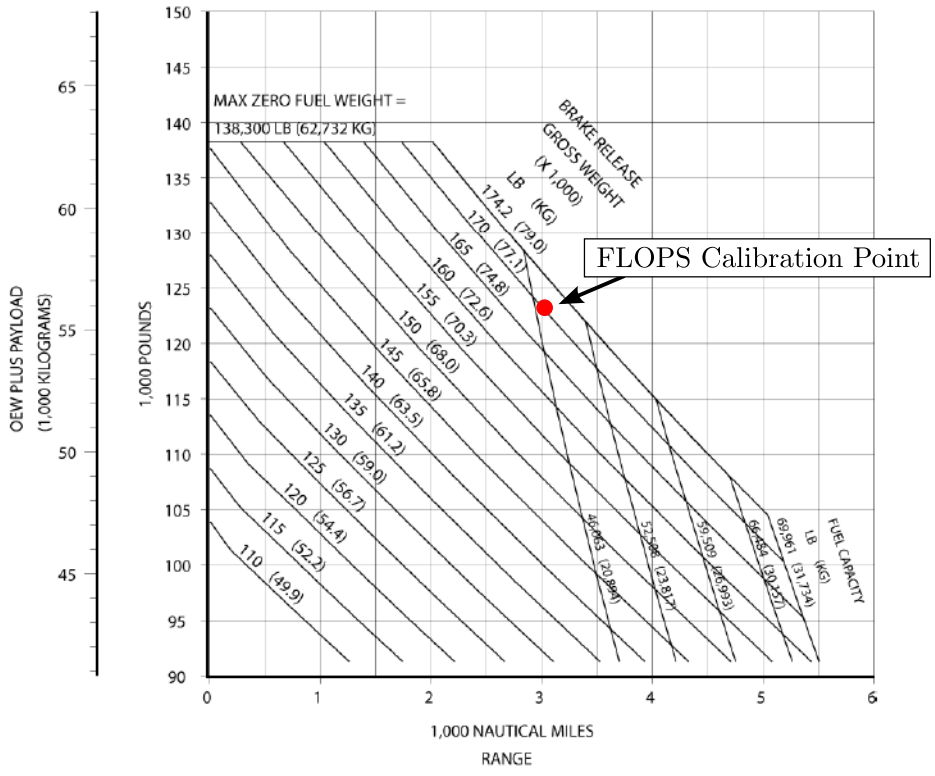


Figure 82: Payload Range Chart for Boeing 737-800 [11]

Once a baseline model was obtained within acceptable error of the reference vehicle, a next generation single-aisle transport could be developed by infusing technologies expected to come online by 2015. This includes heavy use of composites (up to 50%) for the aircraft structure leading to wing, fuselage, and empennage weight reduction of 15%. Cruise Mach was also increased from 0.785 to 0.8 and design range increased to 3250 nmi to reflect performance enhancements. While aircraft geometry remained constant between the baseline and geared architectures, several components required resizing to account for integration issues of large diameter UHB engines. These changes are discussed in the following sections.

6.2.2.2 Landing Gear Sizing

A significant barrier to UHB engines on larger aircraft is installation under the wings. As fan diameter grows, so too must landing gear to avoid ground strike of the larger nacelles. This introduces a weight penalty that must be modeled to obtain accurate system-level results. Three notional engines of varying FPR are shown in Figure 83 to highlight the necessary landing gear changes. The airframe shown is the current 737-800 airframe with baseline engines/landing gear in gray.

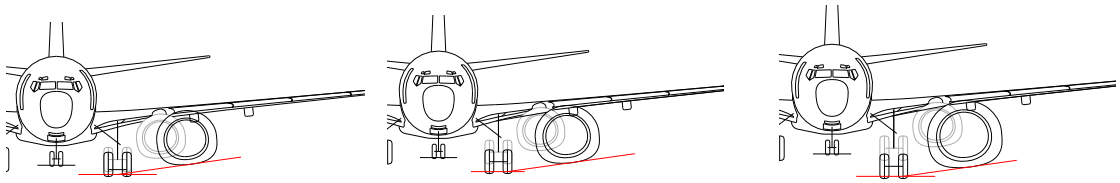


Figure 83: Notional Installation of UHB Engines

Nacelle ground clearance on the 737 is historically small ever since the installation of higher bypass ratio engines beginning on the -300 configuration. In order to accommodate an engine diameter greater than 60 inches, modifications to the landing gear were incorporated into the baseline aircraft model. Two constraints are considered: 1) minimum nacelle ground clearance of 18 inches and 2) nacelle clearance after nose

gear collapse. The landing gear length L_1 required to meet the first constraint is given by Equation 52. These geometries are shown in Figure 84.

$$L_1 = z_{min} + d_{nac} - ((y_{nac} - y_{gear})tan(\Gamma) - \Delta z_{nac}) \quad (52)$$

where:

z_{min} = the minimum ground clearance (ft)
d_{nac} = the nacelle maximum diameter (ft)
y_{nac} = the nacelle spanwise location (ft)
y_{gear} = the main gear spanwise location (ft)
Γ = wing dihedral
Δz_{nac} = nacelle wing offset (ft)

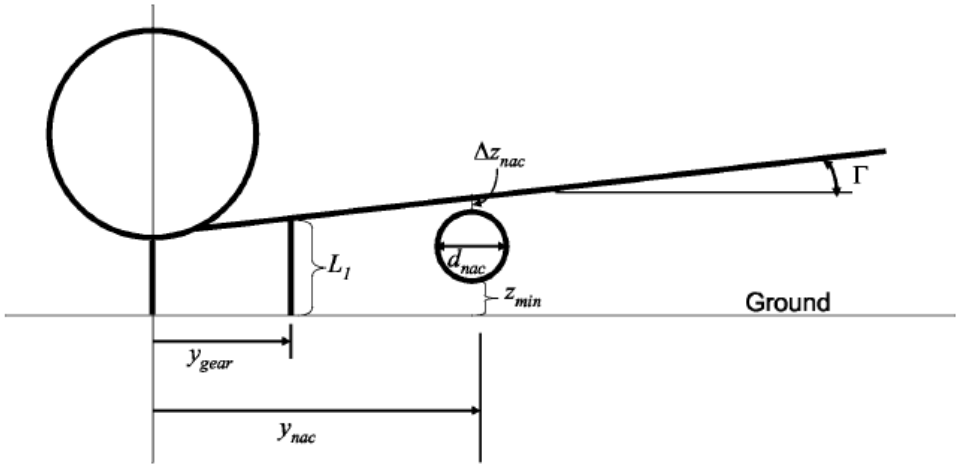


Figure 84: Nacelle Ground Clearance Geometry [66]

The landing gear length L_2 required to avoid damage to the engine in case of front gear collapse is given by Equation 53. These geometries are shown in Figure 85.

$$L_2 = \frac{1}{1-\frac{\Delta x_{nac}}{\Delta x_{fus}}} \times \left[d_{nac} - \frac{\Delta x_{nac}}{\Delta x_{fus}} - \Delta z_{fus} - ((y_{nac} - y_{gear}) tan(\Gamma) - \Delta z_{nac}) \right] \quad (53)$$

where:

Δx_{nac} = horizontal distance between nacelle maximum diameter and main landing gear contact point (ft)

Δx_{fus} = horizontal distance between fuselage contact and main landing gear contact points (ft)

Δz_{fus} = vertical distance between main landing gear (ft) attach point and fuselage contact point (ft)

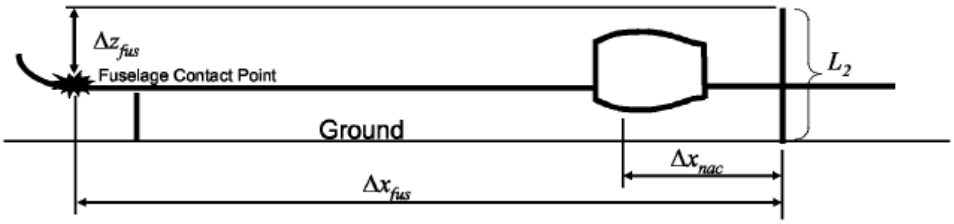


Figure 85: Nose Gear Collapse Geometry [66]

Final main landing gear length was taken as the largest of L_1 and L_2. Nose gear was assumed to be 70% of main gear length. Tire radius was subtracted to obtain landing gear strut length and length was increased 20% to incorporate gear compression.

6.2.2.3 Vertical Tail Sizing

If one engine becomes inoperative during flight, the aircraft must maintain straight and level flight according to FAR 25.149. A minimum control speed must be achieved so the vertical tail can counteract the yawing moment during the engine-out scenario. In addition to asymmetric thrust, windmilling drag of the inoperative engine contributes to yawing moment on the aircraft. As fan diameter increases as with UHB

turbofans, the windmilling drag becomes more significant and tail volume coefficient must be increased to maintain proper stability and control characteristics. A vertical tail sizing routine is incorporated into the model and supplements the FLOPS model for airframe sizing. Two constraints are considered: 1) minimum tail volume coefficient and 2) maximum tail loading during engine-out scenario. The minimum tail volume coefficient was assumed to be equal to that of the baseline 737-800. A minimum vertical tail size $S_{VT_{min}}$ can then be calculated according to Equation 54.

$$S_{VT_{min}} = V_{V_{min}} S_{wing} \left(\frac{b}{l_v} \right) \tag{54}$$

where :

$V_{V_{min}}$ = minimum vertical tail volume coefficient
S_{wing} = wing reference area (ft^2)
b = wingspan (ft)
l_v = vertical tail moment arm (ft)

The tail area required $S_{VT_{oeo}}$ to achieve the necessary sideforce during one-engine-out can be obtained from Equation 55.

$$S_{VT_{oeo}} = \frac{Y_{oeo}}{VL_{max}} \tag{55}$$

where :

VL_{max} = maximum allowable tail loading (lb/ft^2)
Y_{oeo} = required side force given by Equation 56

$$Y_{oeo} = \frac{(T + D_{wm}) y_{nac}}{l_v} \tag{56}$$

where:

T = thrust of the operating engine (lb)
D_{wm} = windmilling drag of inoperative engine (lb)

Estimates of the drag contributions from both external and internal components of a windmilling engine are given by Equations 58 and 57 respectively [168]. Total drag is then given by Equation 59.

$$(C_D S)_{ext} = 0.1\left(\frac{\pi}{4}\right) d_i^2 \quad (57)$$

$$(C_D S)_{int} = \frac{2}{1+0.16M^2} A_N \frac{V_N}{V}\left(1-\frac{V_N}{V}\right) \quad (58)$$

$$D_{wm} = \frac{1}{2}[(C_D S)_{ext} + (C_D S)_{int}]\rho V^2 \quad (59)$$

where:

d_i = engine inlet diameter
M = freestream Mach number
A_N = nozzle area (ft^2)
V_N/V = nozzle velocity ratio
ρ = air density ($slugs/ft^3$)
V = flight velocity (ft/s)

The greater estimate of $S_{VT_{min}}$ and $S_{VT_{oeo}}$ was used as vertical tail area in airframe sizing calculation for FLOPS.

6.2.2.4 Exhaust Emissions

$EINO_x$ models were derived from the NASA Ultra-Efficient Engine Technology program (UEET) and adjusted to reflect advanced combustor technology of a next-generation aircraft [118]. The correlation is given by Equation 60.

$$EINO_x = a_0 \times (P_{t3})^{0.35} \times e^{\frac{T_{t3}-459.67}{300}} \times \left(\frac{f_a}{\Delta\phi}\right)^{2.4} \quad (60)$$

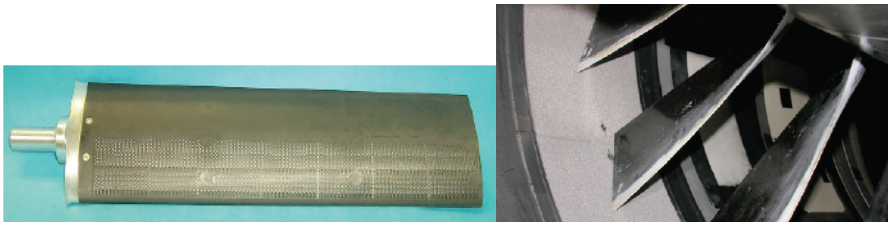
where:

a_0 = function of cooling air and combustor technology level
P_{t3} = compressor exit/combustor inlet total pressure (psia)
T_{t3} = compressor exit/combustor inlet total temperature ($^\circ$R)
f_a = combustor fuel-to-air-ratio
$\Delta\phi$ = combustor cooling air percentage

6.2.2.5 Noise

The analysis tool used to predict cumulative noise for all concepts was NASA's Aircraft Noise Prediction Program (ANOPP) [183]. ANOPP is capable of modeling noise propagation as well as the effects of spherical spreading, atmospheric/lateral attenuation, ground effects, and reflection. The baseline noise model was calibrated with 737-800/CFM56-7B certification data.

Noise reduction technologies were assumed to be applied to the next generation aircraft and engine consistent with entry into service of 2015. Chevrons were applied to all core and fixed-geometry nozzles [81] as well as noise suppression liners on the inlet, interstage and aft fan ducts [119]. Modeling assumptions documented in [66] were made to be consistent with measurements at NASA Glenn's aeroacoustic facility and wind tunnel tests. Other engine noise reduction technologies assumed to be of appropriate readiness for this research were over-the-rotor foam treatment and soft vane stators (shown in Figure 86). These technologies were assumed to improve fan source noise by 4 dB [108]. Airframe noise reduction technologies such as slat cove designs, flap porous tips, and landing gear fairings were also incorporated in the ANOPP model for cumulative noise prediction [53].



(a) Soft Vane

(b) Over-the-Rotor Metal Foam Liner

Figure 86: Fan Noise Reduction Technologies [108]

6.2.2.6 Propulsion System

The basic architecture used across all engine concepts was a two-spool, separate flow turbofan with uniform Aerodynamic Design Points (ADP), the flight condition at which the engine is sized. ADP was chosen as Mach 0.8 and 35,000 ft to represent nominal TOC conditions for the single-aisle transport. A bypass ratio was selected to maintain an extraction ratio of 1.25 (ratio of exit pressures between core flow and bypass flow). In addition to TOC condition, a multi-design point analysis (MDP) was conducted to set flow rates, cycle temperatures and speeds, as well as cooling levels. Fuel-to-air ratio was adjusted to meet a requirement of sea-level static thrust target of 23,000 lb (hot day, +27°F).

The thermodynamic modeling tool used to size the engines was Numerical Propulsion System Simulation (NPSS), an object-oriented, variable-fidelity analysis tool developed jointly by NASA and U.S. industry [35]. NPSS uses a Newton-Raphson solver to converge thermodynamic flowpath properties (pressure, temperature, and speeds) and is seen as state-of-the-art for turbine engine performance modeling. Used in conjunction with cycle modeling is an analysis tool called Weight Analysis of Turbine Engines (WATE) [167]. WATE provides aeromechanical design, flowpath, and engine weight estimates through both semi-empirical and analytical calculations of individual engine components.

The baseline engine was assumed to be CFM56-7B and modeled using publicly available certification data from the FAA and EPA. In predicting performance of a next generation propulsion system, technology forecasts from the FAA's Environmental Design Space (EDS) were used assuming entry into service of 2015. This includes improved materials and efficiencies. Material limits are provided in Table 16. Component efficiencies at ADP are given by Table 17.

Table 16: Next Generation Material Limits

Component	**Material**	**Temperature Limit**
Fan	Polymer matrix composite	N/A
Low Pressure Compressor	Titanium aluminide	N/A
High Pressure Compressor	Titanium aluminide	N/A
High Pressure Turbine	5^{th} generation nickel alloy	3460 °R (turbine inlet, T4) 3310 °R (rotor inlet, T41)
Low Pressure Turbine	5^{th} generation nickel alloy	2460 °R

Table 17: Efficiency Assumptions

Component	**Efficiency**
Fan	Variable (see Figure 87)
Low Pressure Compressor	$\eta_{polytropic} = 0.89$
High Pressure Compressor	$\eta_{polytropic} = 0.91$
High Pressure Turbine (cooled)	$\eta_{adiabatic} = 0.90$
Low Pressure Turbine (uncooled)	$\eta_{adiabatic} = 0.94$
Gearbox	$\eta_{mechanical} = 0.99$

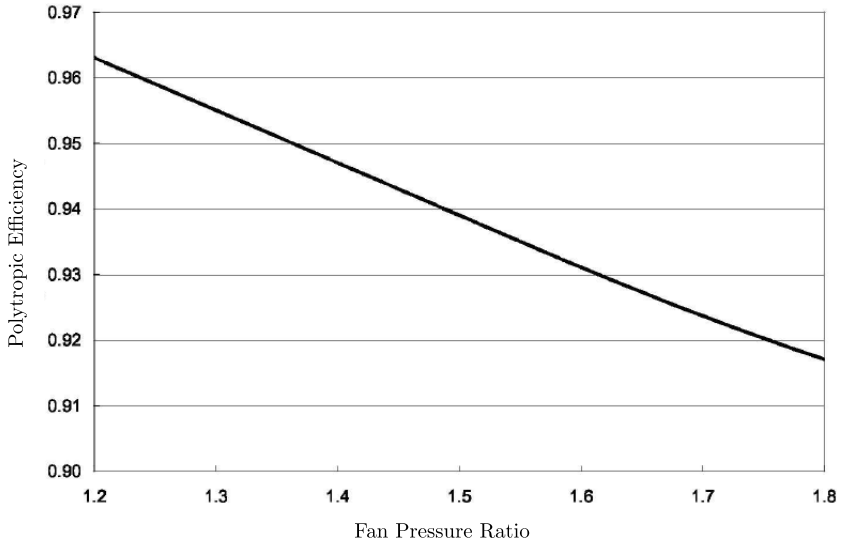


Figure 87: Variation in Fan Efficiency with FPR [66]

Variable area nozzle incurred a 10% weight penalty compared to fixed geometry designs. Nozzle area was allowed to vary in the NPSS solver to maintain peak fan efficiency at off-design conditions.

Table 18: Successful Cases

Concept	**Failure Rate**	**Infeasible Rate**
Direct Drive	21%	31%
Geared/Fixed Nozzle	23%	34%
Geared/Variable Nozzle	30%	34%

Table 19: UHB Model Computational Cost

Concept	**Successes**	**Failures**
Direct Drive	122 sec	86 sec
Geared/Fixed Nozzle	118 sec	81 sec
Geared/Variable Nozzle	121 sec	80 sec

6.2.3 Complexity

The final task of Step 2 involves evaluating the complexity of each model to determine the appropriate evaluation technique for UHB design problem. Basic statistics are provided in Table 18 and 19 for the three concepts. To obtain the failure rates and average computation time for each function call, a Latin Hypercube DOE was run on each concept and for each setting of discrete design variables. This yielded a total of 24 separate DOE runs (eight unique combinations of discrete variables for three concepts). The number of DOE points for each model was chosen to be 40 to reflect the *10k* rule discussed in the previous chapter (where $k = 3$ for the three continuous design variables) as well as provide some additional points in case of failures. These cases will be used as the initial sampling in the PFI-based evaluation methodology.

From the data it can be seen that the unusable cases from a purely random sampling of the space make up between 50 and 60% of all evaluated cases. These numbers could get even worse when searching for optimal designs (not just space-filling) that approach one or several constraints or corner of the design space. It is very common for numerical optimization of real engineering problems to find designs at the extremes of one or more design variables where codes are often the most likely to have convergence issues. Berton et al., in one execution of their multiobjective

optimization using NSGA-II, encountered a success rate of less than 30%.

On average, 2.5 cases must be executed to obtain a single successful design assuming 60% failures and infeasibles. Assuming a worst case where 30% of random cases will fail in 80 seconds and 70% will require 120 seconds, this yields an average computation time for a single feasible case of 270 seconds (four and a half minutes) on an 2.53 GHz Intel Core 2 Duo CPU processor. While the absolute numbers are hardware dependent, the relative computational expense can be compared with the results obtained in Section 5.3.2. Taking into consideration the cost to execute a PFI-based adaptive sampling framework, the cost to locate a new design is orders of magnitude smaller than the cost to run a new point. This fact, along with the tendency for designs to be directed towards feasible regions using PIC sampling criterion, is strong motivation for application of the new concept evaluation methodology to the UHB design problem. The large number of technology assumptions, heuristics, and constraints also make efficiency a priority where a single change in any one could invalidate the analysis.

6.3 Step 3: PFI-Based Concept Evaluation

The methodology continues for the UHB design problem with execution of adaptive sampling scheme based on PIC. UHB engine design is now stated formally as a multi-objective optimization problem in Equation 61. The seven constraints listed in Table 12 are written in canonical form described by Equation 5.

$$\min_{x} \left[W_{fuel}, W_{ramp}, \text{LTO NO}_x, EPNL \right] \tag{61}$$

Subject to:

$$g_1(x) = 1 - D_{TO} \geq 0, \quad g_2(x) = 1 - D_L \geq 0, \quad g_3(x) = 1 - V_{app} \geq 0$$
$$g_4(x) = \dot{h}_{TOC} \geq 0, \quad g_5(x) = W_{EF} \geq 0, \quad g_6(x) = F_{N,SS} \geq 0$$
$$g_1(x) = F_{N,mapp} \geq 0$$

The mapping from the above constrained minimization of four objectives to a maximization problem is shown in Equation 62.

$$\begin{aligned} \max_x \{ MPPI \cap P[D_{TO} < 7,000] \cap P[D_L < 7,000] \cap P[V_{app} < 150] \\ \cap P[\dot{h}_{TOC} > 300] \cap P[W_{ef} > 0] \cap P[F_{N,SS} > 0] \\ \cap P[F_{N,mapp} > 0] - NPD \} \end{aligned} \tag{62}$$

or, assuming independence

$$\begin{aligned} \max_x \{ MPPI \times P[D_{TO} < 7,000] \times P[D_L < 7,000] \times P[V_{app} < 150] \\ \times P[\dot{h}_{TOC} > 300] \times P[W_{ef} > 0] \times P[F_{N,SS} > 0] \\ \times P[F_{N,mapp} > 0] - NPD \} \end{aligned} \tag{63}$$

The feasible cases executed for complexity investigation were used as the space-filling points to seed the design space and kick-start the adaptive sampling portion of Step 4. Kriging surrogate models were fit through all four objectives and seven constraints yielding a total of 11 surrogates per concept. The four unique combinations of discrete variable settings and three independent concepts brought the total number of Kriging models to be trained to 132 ($11 \times 8 \times 3$). Considering the fact that Kriging is notoriously expensive, this may seem like a severe disadvantage to PFI-based sampling where the overhead cost of training surrogate models dominates the total computational budget. However, training each model after the initial sampling took on the order of seconds with the relatively small sample size. Additionally, subsequent surrogate updates were minimized by training only once after the initial

Table 20: Genetic Algorithm Settings

Parameter	**Value**
Selection Method	Tournament
Elitism	True
Generations	5
Population	100
Crossover Rate	0.9
Mutation Rate	0.1
Variable Encoding	Real
MCS Samples	150

sampling. It is important to note that "training" constitutes re-optimizing the hyperparameters. If system behavior is sufficiently well-behaved, it is not expected that these parameters will vary greatly during the iteration (see Figures 69 and 70). Experimentation will determine if this assumption is appropriate. If training is required at every new design (or every n^{th} design), it is assumed a gradient-based optimization of the hyperparameters with initial guess carried over from the previous training will converge very quickly.

For this problem, computational work was spread evenly between initial and adaptive evaluations. Using a $(10k + failures)$ rule, this yielded 40 space-filling and 40 adaptive samples for each concept and discrete setting. This equated to 320 function evaluations on each model and 960 total function calls across all concepts and discrete setting. With each new adaptive sample, 12 separate optimizations (three concepts and four discrete settings) were performed to locate the designs that maximized PIC. A genetic algorithm was used to perform global optimization of the highly multimodal PIC function in three dimensions (wing area, SL Thrust, and FPR). The settings for the GA implementation Python library PyEvolve are summarized in Table 20.

6.4 Step 4: Data Visualization and Analysis

The final step of the PFI-based evaluation methodology aims to draw conclusions from the results obtained in Step 3 and ultimately facilitate concept selection decisions.

This section is divided into four general areas of focus:

- Visualizing the Pareto Hyperspace - visualization remains one of the most difficult aspects of multiobjective decision making. Multivariate scatterplots offer a way to draw conclusions quickly about complex relationships between high-dimensional sets of data. This approach will enable a discussion of UHB performance and study the impact of design variables on the objectives. General comments and observation about UHB engine design will be presented that are largely independent to the method used to evaluate each concept.
- Method Robustness - with many conceptual design tools, the rate at which failed or infeasible designs are evaluated is often high enough to justify the overhead cost of PFI-based evaluation. This section will investigate the performance of the proposed methodology at locating feasible designs and benchmark against random sampling. The benefits of the methodology will be shown from the perspective of robust sampling.
- Characterizing Infeasible Regions - those cases that failed or violated one or more of the constraints will be examined for the underlying causes. The proposed sampling procedure will be further evaluated for the ability to balance search for optimal designs with probability of feasibility and convergence.
- Characterizing Technology Tradeoffs - the ultimate goal of the PFI-based methodology is to efficiently unearth properties of competing concepts that cause a shift in preference from one technology alternative to the next. The Pareto Distance measure presented in *Experiment 3.2* is used here to uncover impact of the design variables on Pareto optimality as well as investigate similar performing optimal concepts.
- Benchmarking - sampling performance of PFI-based evaluation will be compared with two other evaluation methods: sequential (MOPI) and simultaneous optimization (MPPI). The ability for the method to reduce predictor area

in regions of optimality shift will be analyzed and contrasted for the various methods.

6.4.1 Visualizing the Pareto Hyperspace

The analysis step of the PFI-methodology now continues with a general discussion of the performance characteristics of UHB engines and their dependence on the design variables. Of the six unique pairs of objectives, four relationships proved to have the most significant trades: ramp weight/fuel weight versus noise/NOx. This left the noise/NOx relationship and ramp/fuel weights as having a high positive correlation. In general, if one of these objectives was minimized, so was the other. The correlations, calculated from all successfully evaluated designs, are listed in Table 21 (bold indicates strong positive correlation). From an engine design perspective, lower fan pressure ratio leads to lower bypass stream exhaust velocity which significantly lowers noise. Lower FPR also leads to high BPR and thus more thrust from the bypass stream during takeoff. Under the current set of modeling assumptions, bypass ratio is independent to engine architecture or other design variables other than FPR. This is shown in Figure 88. For this reason, lower FPR leads to decreased emissions and explains the strong correlation between noise and LTO NOx. The correlation is further emphasized in the plots of FPR versus these two objectives shown in Figure 89. The three concepts are plotted as different colors in the scatterplot. For ease of visualization, only the adaptive samples are shown. The effect of the discrete design variable OPR can also be seen in the bottom portion of Figure 89. The strong positive correlation between fuel weight and takeoff weight is shown in Figure 90. The design variables of wing area and thrust act as scaling parameters for the aircraft and engine. Larger wing and higher thrust engine lead to heavier structure and thus more required fuel to accomplish the prescribed mission.

Strong positive correlation among several objectives simplifies the optimization

Table 21: Objective Correlations

	Wf	Cum Noise	Wr	LTO NOX
Wf	1.0	-0.0080	**0.9373**	-0.1529
Cum Noise	–	1.0	-0.2460	**0.8308**
Wr	–	–	1.0	-0.2607
LTO NOx	–	–	–	1.0

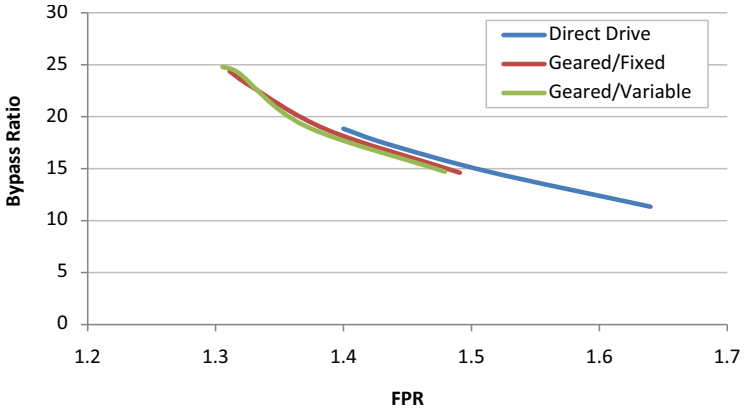


Figure 88: Bypass Ratio Variation with Fan Pressure Ratio

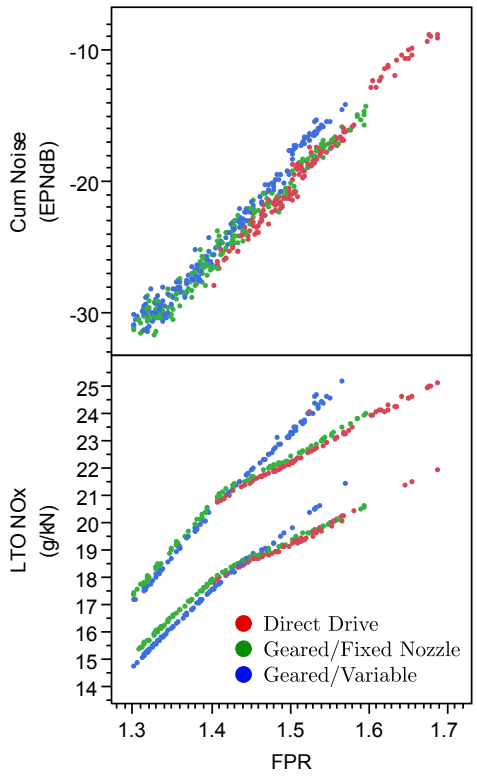


Figure 89: Noise and Emission Variation with FPR

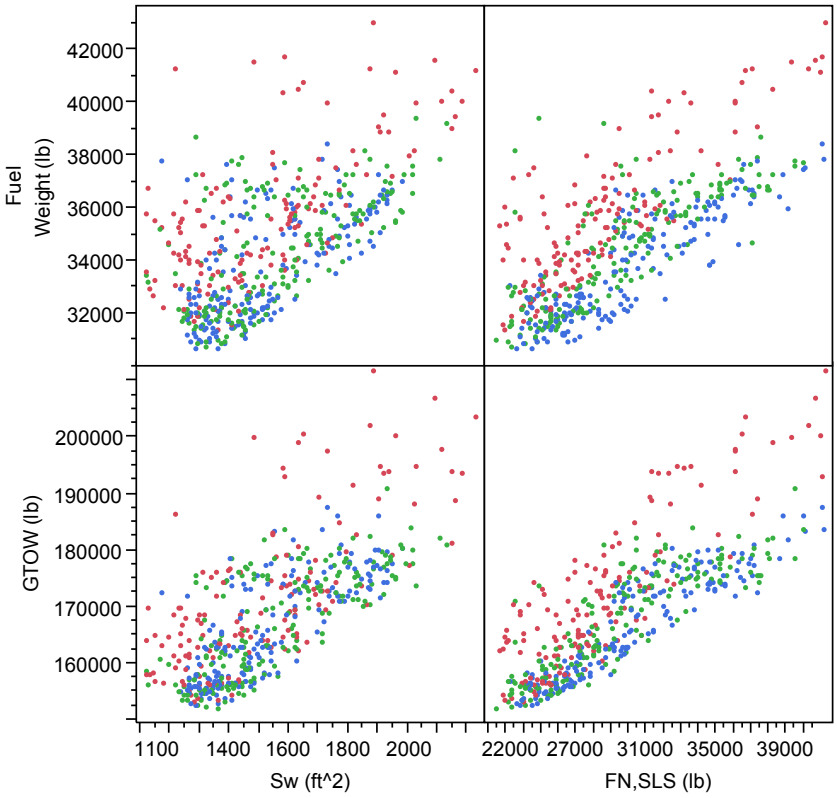


Figure 90: Weight Variation with Thrust and Wing Area

problem slightly as well as visualizing the Pareto frontiers in multiple dimensions. However, the tradeoff between weight and environmental metrics still dictates that a compromise solution must be attained.

The objective tradeoffs are plotted as two-dimensional projections in Figure 91. The three competing concepts are shown as different colored points. Adaptive samples are depicted by open circles, and initial samples are depicted by small dots. For clarity, only the individual Pareto frontiers for each concept are shown. While many designs appear to be dominated in the two-dimensional projections, each point is in fact Pareto optimal for that particular concept. This highlights the difficulty with establishing Pareto optimality from visual inspection of multivariate scatterplots. For problems in higher dimensions and less correlation between objectives, visual determination of Pareto optimality is even more cumbersome.

The two correlated objectives, ramp and fuel weights as well as NOx and noise, can be seen in the figure. The collection of samples for these objectives is localized about a line with positive slope indicating simultaneous minimization from a single design. Both noise and LTO NOx are reduced as fan pressure ratio increases and explains the strong positive correlation between these objectives. While OPR is the same at the aerodynamic design point, takeoff OPR decreases with lower FPR leading to lower emissions. It should be noted that the study focused on reducing the FAR emissions parameter which is normalized by engine sea-level thrust and does not necessarily lead to total emissions reduction. The differences in required thrust from a larger fan and lower FPR engines are not captured by this parameter. Another observation is the impact of the discrete variable, OPR. While designs at *both* high and low overall pressure ratios are Pareto optimal, it is difficult to determine from the plots in the far right column where designs with high OPR appear to be dominated. OPR directly sets compressor exit flow properties, P3 and T3, and explains the behavior seen in the middle-right scatterplot since the emissions prediction is highly correlated to these

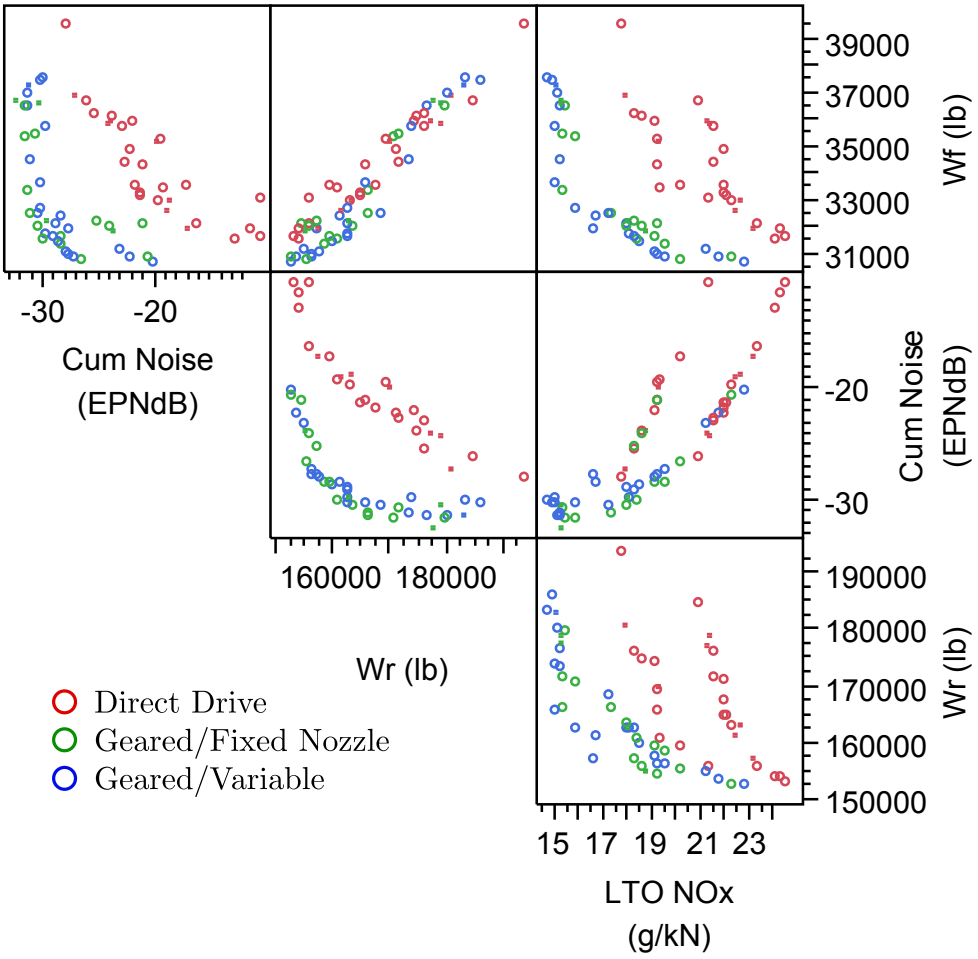


Figure 91: Competing Concept Pareto Frontiers

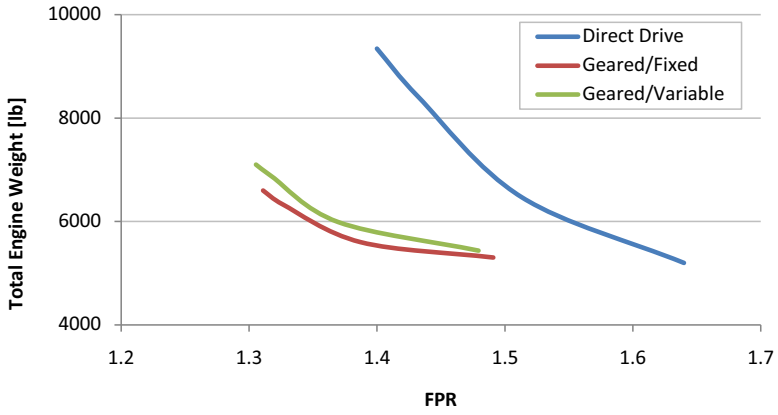
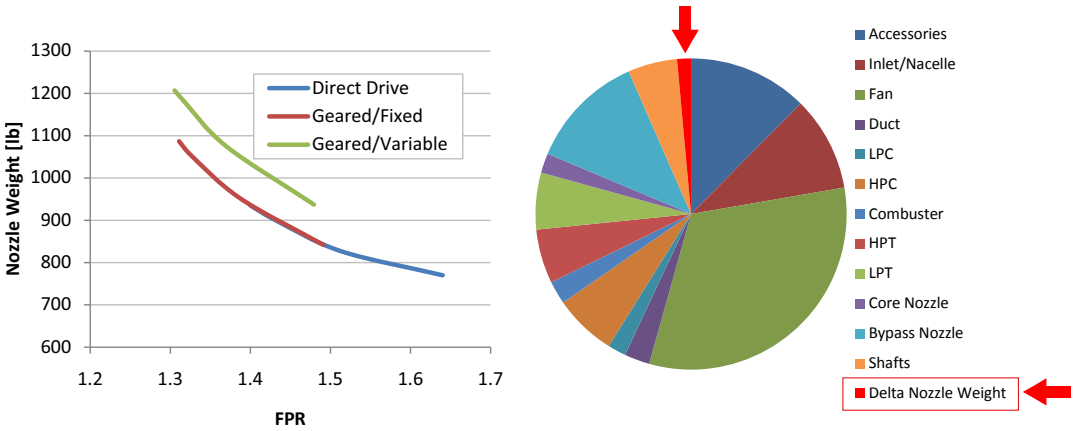


Figure 92: Total Engine Weight Variation with FPR

thermodynamic cycle parameters (see Equation 60). As OPR varies from low to high, the LTO NOx emissions are basically shifted to high regions.

In terms of optimality among concepts, the direct-drive architecture is dominated completely by the gear-driven designs along every objective. A conclusion to draw from this fact is the performance benefits afforded by a gear outweigh the weight penalty associated with its installation on UHB engines. For a given FPR and loading, a gear reduces *overall* engine weight by reducing number of turbine stages and simplifying low pressure compressor design. This is seen in Figure 92. As FPR increases, the weight difference between geared and direct-drive narrows as compressor/turbine stage count can be reduced for the direct-drive architecture. While the other design variables associated with engine sizing (OPR, work split, thrust) can significantly contribute to engine weight estimation, the weights presented in Figure 92 are meant to give general trends in behavior for the three concepts.

Another important result is no obvious preference toward variable or fixed area bypass nozzle. This is explained by the fact that the objectives of interest in this study are not able to capture with enough resolution the improved performance of engines with variable geometry. Additional objectives such as maintenance cost or reliability metrics may be necessary to inform decisions between the two architectures. For this



(a) Nozzle Weight vs. FPR (b) Engine Weight Breakdown

Figure 93: Nozzle Weight Results

analysis, the weight penalty of a more complex nozzle is confounded by the fact that nozzle weight simply accounts for too small a percentage of overall engine weight to manifest in the system-level metrics laid out by NASA FAP. This is illustrated in Figure 93. The penalty of a variable area nozzle is a little over 100 pounds as seen in Figure 93a. However, this delta accounts for a small percentage of total engine weight. The weight breakdown for a representative variable area nozzle engine is shown in Figure 93b.

An important trade identified in [23] is the relationship between ramp weight and noise. In fact, due to the strong correlation between ramp and fuel weight (as well as NOx and noise), the four-objective space can be easily visualized by the single tradeoff space shown in Figure 94. The performance of the geared fans is more easily seen in the figure. While still completely dominated, only for low values of ramp weight does direct-drive architecture become remotely desirable.

A general conclusion that can be made is that as fan diameter (bypass ratio) decreases, the weight benefits from a gear decrease. LPT stage count can be sufficiently reduced at the same time maintaining reasonable turbine loadings. The weight savings from decreased part count make direct-drive engines lighter overall. The geared

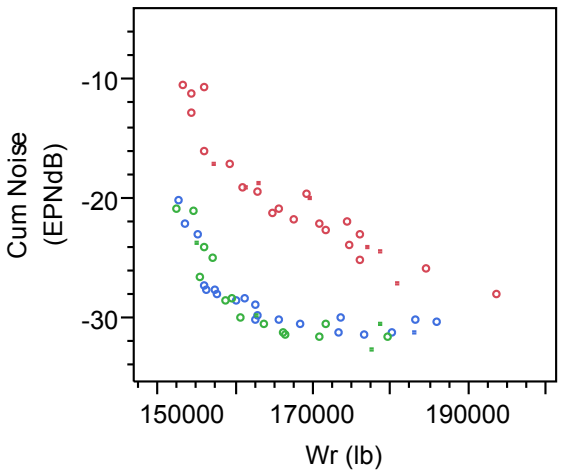


Figure 94: Ramp Weight vs. Noise - Pareto Points

architecture still dominates along the noise objective, but before preferences are specified, the minimum weight/maximum noise designs must still be considered.

The next results presented investigate how the engine changes as preferences on the objectives change. In other words, designs are studied as they move from best-in-class, to compromise solutions, to the opposite extreme along the Pareto front. Four representative engines are chosen for each architecture, beginning with minimum weight/maximum noise, and are evenly spaced along their individual Pareto optimal set. The engine cross-section provided by WATE is shown along with its corresponding placement along the Pareto frontier. Direct-drive engines are shown in Figure 95, geared/fixed in Figure 96, and geared/variable in Figure 97.

Even upon cursory observation of the direct-drive architectures, it is easy to see why a geared fan offers such significant weight savings. For very low fan pressure ratios, the stage counts in the low pressure compressor and turbine quickly become unreasonable. The lightest Pareto optimal engine has a 5-stage LPT and FPR of 1.64, on the high end of the allowed range. The heaviest engine and minimum FPR design has a 14-stage LPT that almost doubles the weight of the engine. The increased stage-count lengthens the engine by approximately 70 inches. Under the current set

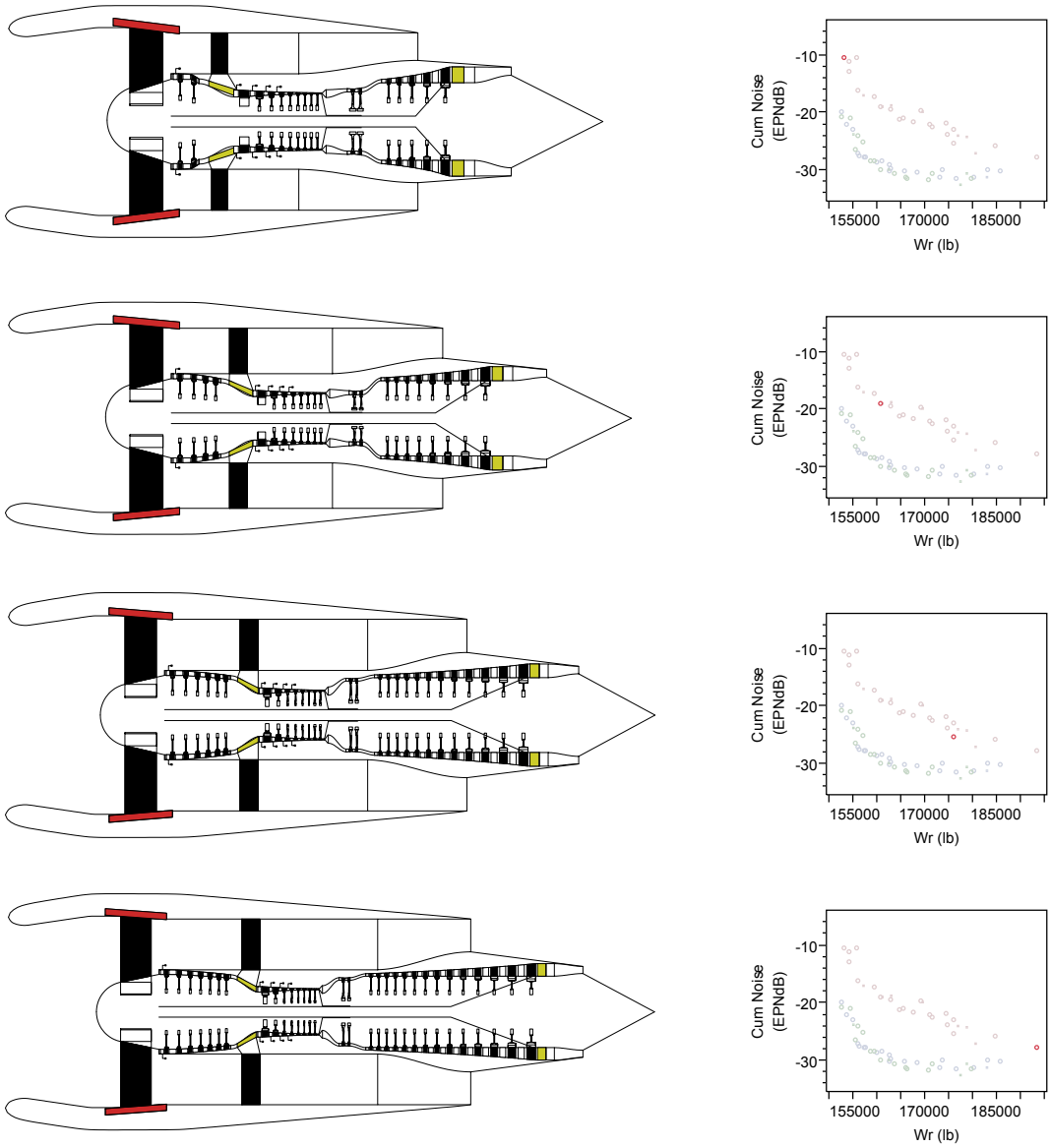


Figure 95: direct-drive Pareto Optimal Designs

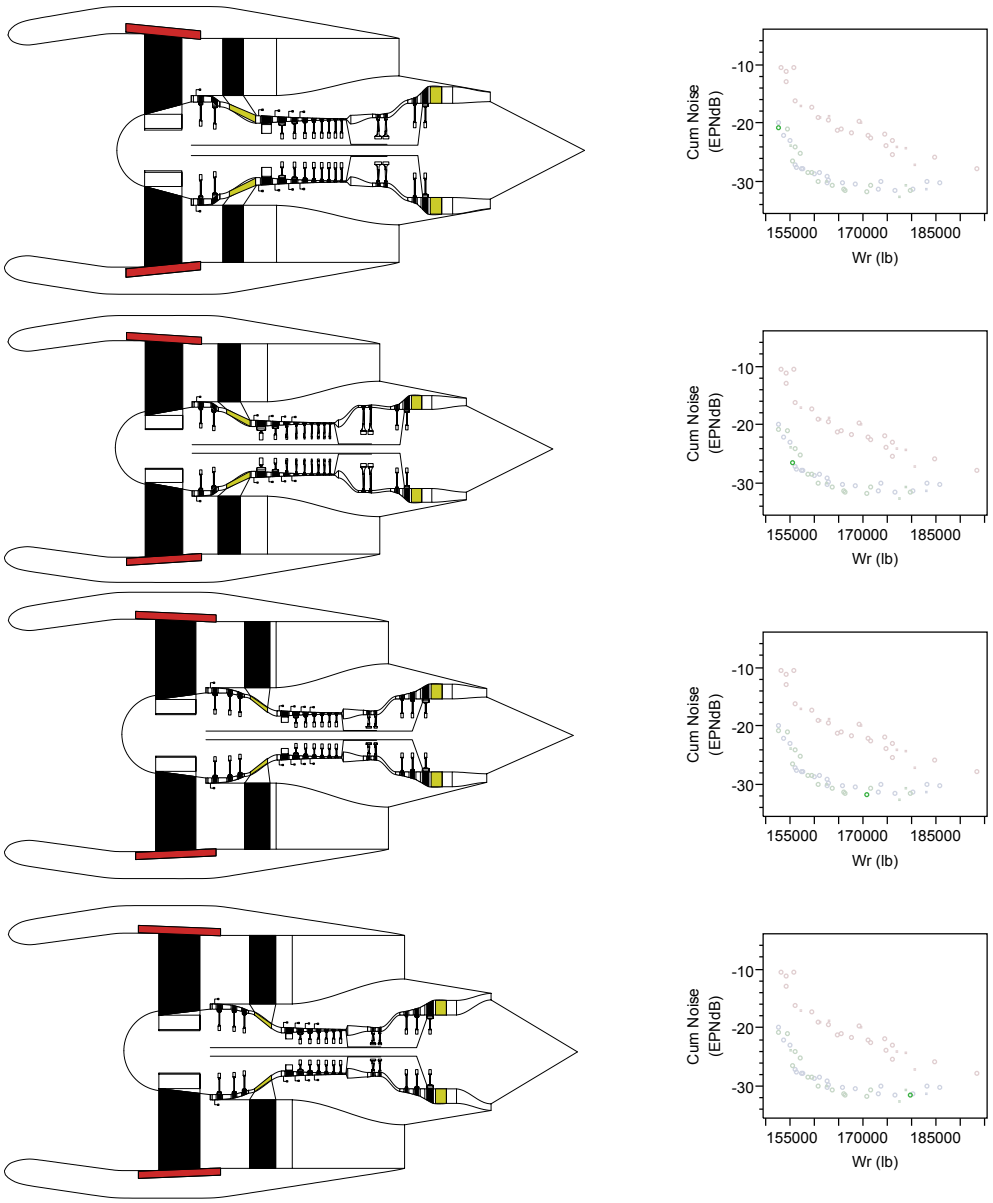


Figure 96: Geared/Fixed Pareto Optimal Designs

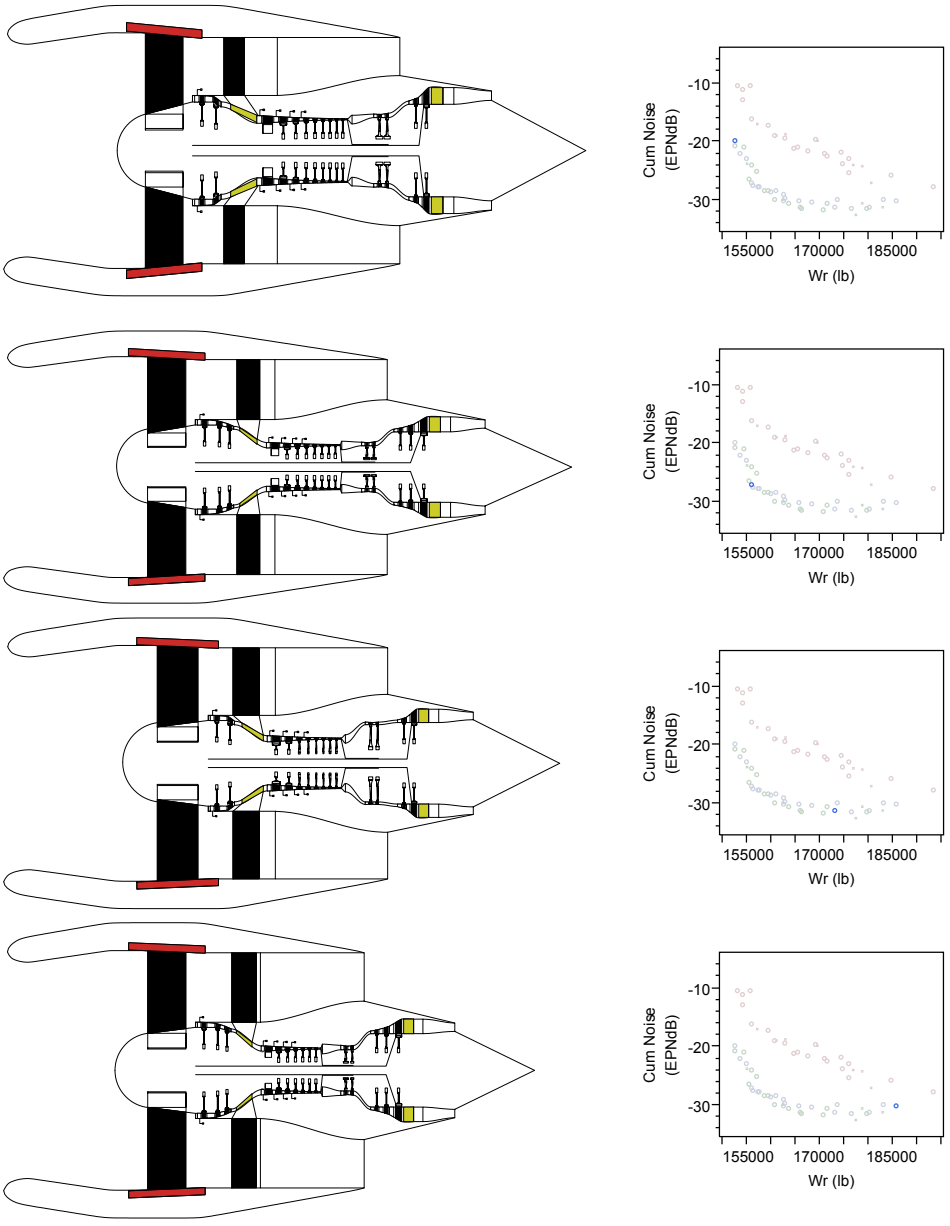


Figure 97: Geared/Variable Pareto Optimal Designs

of assumptions (constant fan loading, higher turbine inlet temperature), lower fan pressure ratios produce almost nonsensical direct-drive engines.

The flexibility in turbine design can be seen from the geared fan cross-sections in Figures 96 and 97. LPT stage count remains between two and three as fan pressure ratio decreases to its minimum allowed value of 1.3 for the geared architectures. Compared with direct-drive, stage count is much less sensitive to fan pressure ratio because gearing allows for LPT rotational speed to vary independent of fan speed. Such a decrease in stage count on the LP-spool significantly reduces overall engine length but increases engine diameter which has further implications on installation. In general, as designs traverse the Pareto front from high noise/low weight to low noise/high weight fan pressure ratio decreases. Thrust and wing area of the Pareto designs must increase to accommodate the heavier engine and satisfy the mission requirements.

6.4.2 Method Robustness

The results obtained above are largely independent of the particular analysis method used, and support the findings from prior UHB design studies. If the designer is willing to pay the computational expense, evolutionary computation or even random sampling can eventually provide characterization of the design space. The purpose of this and the following sections is to demonstrate efficiency improvements of the proposed method.

The results presented here will show the robustness of the PFI-based methodology. That is, the ability of the method to locate successful, feasible cases compared with traditional methods. Two sampling methods are compared: purely random and the adaptive samples from PFI iteration. The purely random samples were taken from the initial set of training data, 160 points for each concept. It is assumed the Latin Hypercube DOE is adequately space-filling and a larger sample size would yield

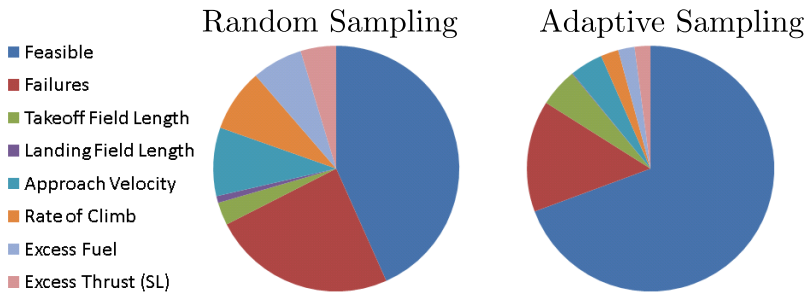
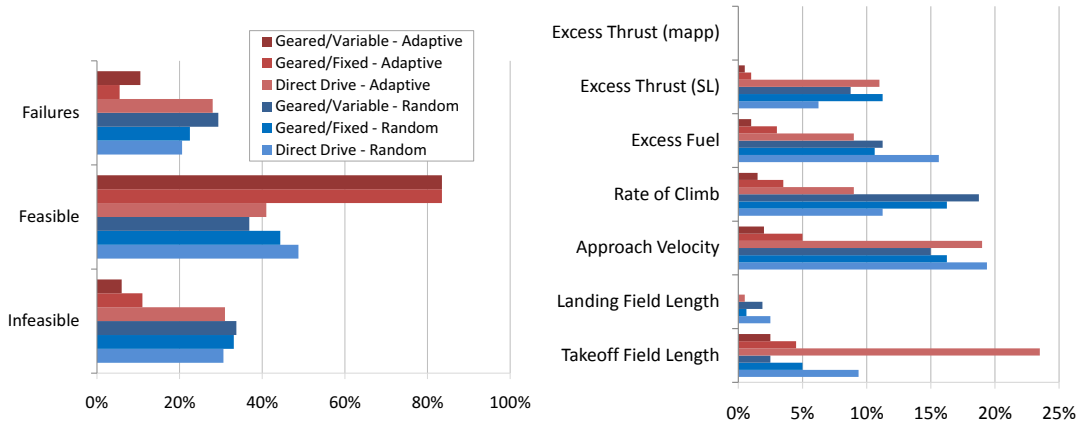


Figure 98: Feasible Point Comparison

similar statistics. The results for all three concepts are shown in Figure 98. Compared with purely random sampling, there are significantly more feasible evaluations across all concepts with reduced frequency of failures and infeasible points. 70% of the adaptive samples were feasible compared with only 43% from the initial population of designs. The number of failed cases was also reduced from 24% to 15%. These results are especially promising considering the adaptive samples are placed in areas of optimality where constraints and invalid inputs are more likely. This is common to many numerical optimization routines, where infeasible regions are in close proximity to optimum solutions.

The feasibility statistics are further broken down by concept and constraint in Figure 99. Figure 99a shows each concept and percentage of points that are either feasible, infeasible or failed. The geared architectures show significantly more successful evaluations than the direct-drive designs. The ability to locate feasible points is actually decreased for direct-drive concepts, but the relative amount of total feasible cases is still increased. The reason for this can be seen by examining the Pareto frontiers in Figure 91 and considering the two components of the PIC infill criterion. The same phenomenon was observed in the completely dominated truss designs from the example presented in Section 5.3. When there is very low probability of improvement over all concepts, as with the case for completely dominated concepts, MPPI drops below machine precision. The search then proceeds with locating those points that



(a) Sampling Results by Concept

(b) Breakdown of Constraints

Figure 99: Robust Sampling Results

minimize NPD. For unconstrained optimization with relatively robust models, this is less of a concern as minimizing the distance to known Pareto optimal designs should place designs on a Pareto frontier of a single concept. However, in situations such as the engine design problem that are highly constrained and exhibit complex failure modes, this leads to significantly more failed cases. The problem is exacerbated when the areas of optimality overlap the infeasible regions.

The infeasible points are broken down into the seven constraints in Figure 99b for the initial and adaptive samples. The percentages are relative to the total number of infeasible points for that concept and so do not add up to 100 percent. In other words, the bar chart double bookkeeps cases that violated more than one constraint. For the design variables and assumed ranges, the rate of climb and approach velocity requirements are the most stringent. The large number of infeasible points for the direct-drive concept is reflected here as well.

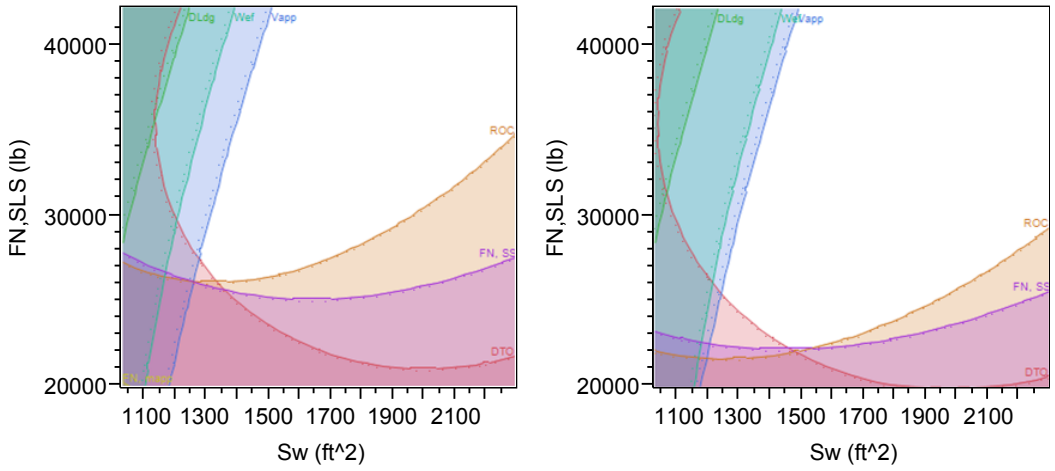
6.4.3 Characterizing Infeasible Regions

Generally, there are two ways a single evaluation in any modeling code can fail. The first is a truly random occurrence and can be caused by numerical instability, rounding error, or other unforeseen and difficult to predict phenomenon. Addressing

these types of failures is beyond the scope of this thesis. The second type of failure and one that is specifically dealt with herein, is caused by specific properties of the design space. Typically encountered in extreme values of acceptable design variable values (or combinations of multiple variables), these failures can be correlated directly with the inputs. The PFI-based evaluation seeks to avoid failures of this type by artificially increasing the objectives above what is predicted by the Kriging surrogate models. In minimizing the objectives, the iteration simultaneously avoids potential failure areas.

Similar to failures, infeasible points most often are a result of a particular property of the input variables. The mission constraints imposed on the system carve out irregularly shaped areas of the design space. For many DOE and evolutionary optimization algorithms, these areas can be difficult to avoid and may lead to a large amount of wasted computation. From the statistics presented in the previous section, adaptive sampling already does a better job relative to the "zero overhead" methods at finding feasible points. The purpose of this section is to investigate the root causes of those constraint violations and define the boundary between feasible and infeasible in greater detail.

The design space is first presented in terms of wing area and thrust, two of the most significant sizing parameters for conceptual aircraft design. Figure 100 shows these two design variables and differs from the classical "constraint diagram" only in that the axes are not normalized by takeoff gross weight. The constraints are also shown as shaded areas on the plot and represent certain combinations of wing area and thrust that produce an aircraft that cannot meet all of the performance requirements. As FPR, the other continuous design parameter, is allowed to vary across its acceptable range, the behavior of the constraint diagram is altered. This is captured in the side-by-side comparison of Figure 100a and 100b. A collapse of the feasible design space can be seen as FPR decreases from its maximum value (1.7) to its minimum (1.3). Another important note is that the constraints most closely tied



(a) Low FPR (b) High FPR

Figure 100: Constraint Diagram for the UHB Design Problem

to thrust (excess thrust, rate of climb, and takeoff field length) are most affected by changes in FPR compared with those constraints more impacted by aircraft geometry (landing field length, excess fuel, and approach velocity). This can be explained by the impact of thrust lapse, or how thrust tapers off with Mach number and density. For higher bypass, lower fan pressure ratio engines have a lower specific thrust where thrust drops off more rapidly with Mach number. The feasible design space is larger for smaller fan, higher specific thrust engines that are less susceptible to thrust lapse than low specific thrust engines.

Sampling from the PFI iteration is now presented in terms of the same design parameters in Figure 101. Both initial (closed shapes) and adaptive (open shapes) samples are shown in the figure. When comparing the shaded constraint diagram with the evaluated designs from the concept models, the reason for constraints violation can be seen. Regions of extreme low wing area and low thrust are infeasible and are consistent with the above shaded areas. Another observation is the difference between initial and adaptive samples. The majority of the adaptive samples that failed were placed in low wing area, low sea-level thrust design space. The reason for this is that samples were placed here to minimize ramp weight and fuel weight, which

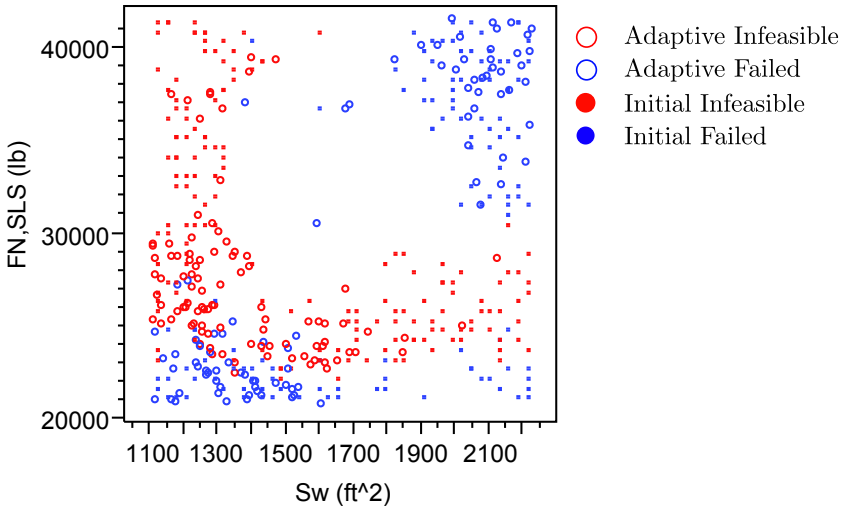


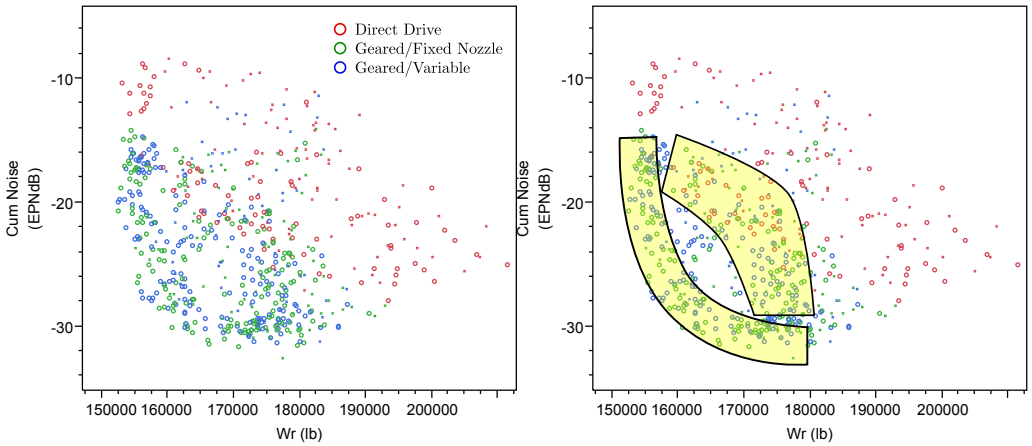
Figure 101: Infeasible Sampling

have high positive correlation with wing area and thrust. In searching for Pareto optimal points, the algorithm nevertheless placed samples in infeasible areas. This is an important result as PFI-based sampling balances search for feasibility with search for optimality.

Two distinct regions of failed cases can be seen in the plot of wing area versus thrust. The first type of failure is due to a negative excess thrust during the second climb segment of the prescribed mission. As expected, a low sea-level thrust triggers this failure (blue dots along the bottom of Figure 101). Another failure mode is encountered by designs of high wing area and high thrust, the top left cluster of blue designs. Cases fail in this region due to the noise prediction program, ANOPP, failing to converge.

6.4.4 Characterizing Technology Tradeoffs

The major contribution of PFI-based evaluation is presented in this section. To investigate the sampling performance of the methodology, a two-dimensional scatterplot of ramp weight versus noise is shown in Figure 102. Initial space-filling samples are plotted as small closed circles, and adaptive samples are open circles. The most significant feature in the figure is the clustering of geared designs (blue and green circles)



(a) All Samples (b) Sample Clustering

Figure 102: Ramp Weight vs. Noise Tradeoff

along the Pareto frontier. The four-objective minimization problem has been reduced to a projection in two dimensions. This partially explains the thickness to the band of designs near Pareto optimality. In searching for optimality across *all* objectives, some designs necessarily fall in areas of apparent suboptimality when projecting the data into two dimensions. Another reason for the thickness is due to the error in the surrogate prediction and optimization. Designs were located based on the expectation of improving, which may deviate significantly from true improvement.

Closer inspection of Figure 102a reveals two distinct "bands" of clustered samples. These are highlighted in yellow in Figure 102b. The reason for the clustering is explained by examining the functional form of the infill criterion (given by Equation 63). Each cluster of designs corresponds to a term in the PIC formulation. Designs near the Pareto frontier are found from maximization of the $MPPI$ component of PIC. The second cluster consists of those designs that maximize the NPD portion of the infill criterion. These points are suboptimal because they are forced to be similar in performance to Pareto optimal designs of direct-drive architectures, the concept that is completely dominated across the design space.

The band of points corresponding to NPD maximization is more easily seen in

Figure 103. Geared designs, both variable and fixed geometry nozzles, are colored according to their Normalized Pareto Distance from the direct-drive Pareto frontier: dark green indicates closer proximity to the Pareto frontier of direct-drive engines. The designs that are optimal from the perspective of only direct-drive engines are plotted as red points. It can be seen that the band of points is centered around this optimal set with the lowest Pareto Distance. It can be argued that these designs were wasted computation because they are not Pareto optimal. However, the dark designs are those geared configurations that are closest in performance to the best performing direct-drive engines, the ultimate goal of PFI-based evaluation. It just so happens that these designs are dominated by design from other concepts. A conclusion to draw from this is that the PFI-based evaluation can have difficulty balancing the two objectives, optimality (and feasible) with similar performance between designs. For the algebraic sample as well as truss design problem, both objectives could be maximized concurrently because dominance shifted from concept to concept as objective preference varied yielding distinct Pareto intersections. For the more complex problem of constrained engine design, the relationship between geared and direct-drive architectures is not so well-behaved. Complete dominance by geared architecture causes samples to alternate between optimal and suboptimal creating two distinct bands of clustering.

The alternation between optimal and similar performance can also be seen by examining the iteration history of Pareto Distance for the adaptive samples of the three concepts. Figure 104 shows direct-drive (red dots), geared/fixed (green dots), and geared/variable (blue dots) for the 40 iterations of PFI evaluation. The first observation is the overall lack of trend in the data. Ideally, samples would decrease in Normalized Pareto Distance as surrogate accuracy increases and designs are driven closer to the performance of other concepts. However, due the dominated nature of the direct-drive architectures and lack of preference between variable and fixed

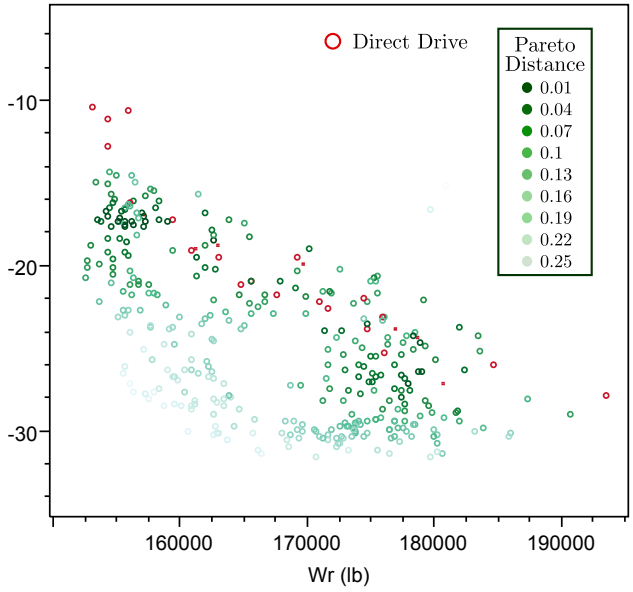


Figure 103: Distance Between Geared and Direct-drive Pareto Designs

nozzle, there is an almost random scatter to the iterate. This supports the conclusion that designs are placed almost alternatingly between the s-Pareto frontier and similar performing, although suboptimal designs. Another observation that can be made from examining Figure 104 is the behavior of the direct-drive engines (red dots) as the iteration progresses. In fact, Pareto Distance got worse (even more random) beyond the tenth adaptive sample. This is explained by two phenomena. First, the portion of the infill criterion designed to locate optimality, MPPI, has exploited knowledge of the best possible designs and shifted towards exploration. In exploring unknown regions, some designs are evaluated that are well off the Pareto frontier. The modeling uncertainty given by the Kriging predictor is large enough that improvement upon the current set is probable. The second cause is that these cases are also much more likely of being feasible which dominates the infill criterion calculation.

As mentioned in Section 6.4.1, decreasing FPR, with corresponding increases in wing area and thrust, traverses the Pareto frontier from low to high ramp weight aircraft. As FPR increases, the advantages of using a gear-driven fan diminish as

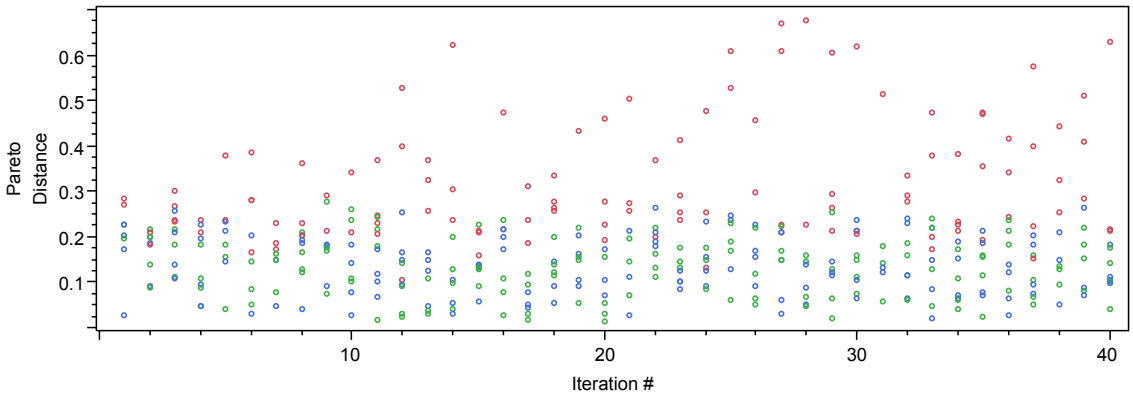


Figure 104: Pareto Distance Iteration History

direct-drive technology can sufficiently lower stage count to achieve lower weight. Another way of looking at this is by inspecting the gear ratio as FPR increases. A FPR of 1.6, the highest allowable value, yields an optimum gear ratio of around 1.2. Clearly, as gear ratios approach one, the benefits of the reduction in RPMs between LPT and fan are counteracted by the weight penalty of carrying a gearbox. The key question posed at the beginning of the evaluation has in a sense already been answered. *Where does optimality shift from one concept to another?* Under the current set of modeling assumptions, a gear driven fan is always optimal for any combination of objective preferences.

What is important here is that the algorithm attempts to find direct-drive designs that are *most likely* Pareto dominant. This is seen from the cluster of direct-drive adaptive samples (red circles) in the region of low ramp weight and high noise, where the probability of improving over the current set of designs is highest. The sampling criterion guided designs into this region because of a maximization of the probability of being dominant, the MPPI component of PIC. For the geared architecture, a cluster of designs can be seen in Figure 103 that is both 1) close to direct-drive engines and 2) close to Pareto optimal. These are the designs of low ramp weight and high noise that correspond to high FPR engines. The cluster is highlighted in

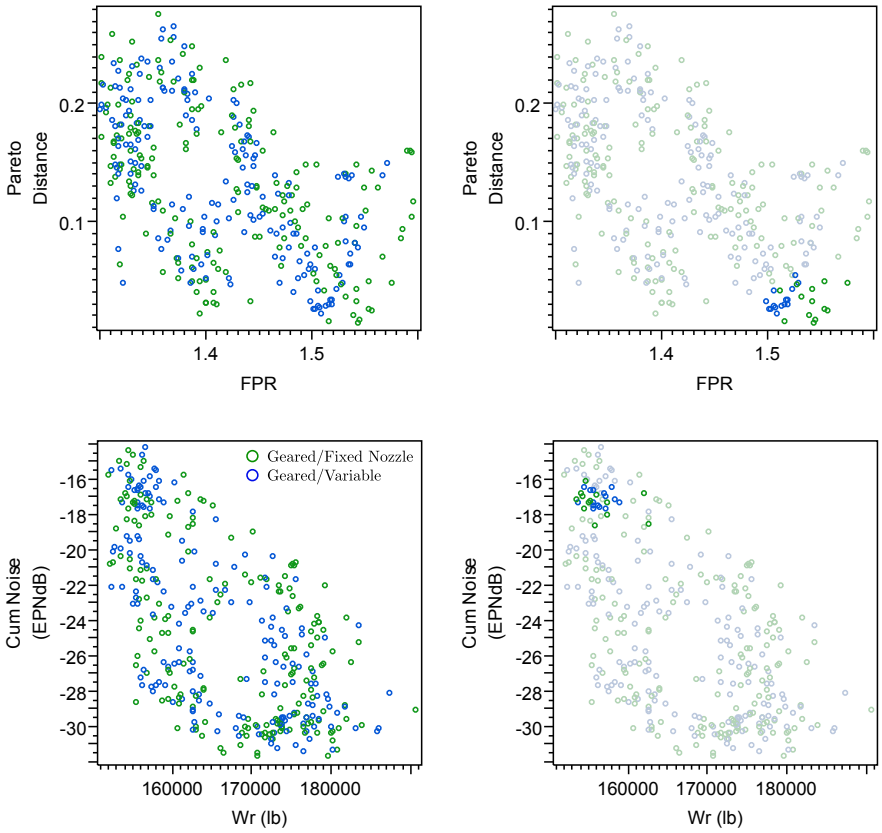


Figure 105: Locating Optimality Shift Between Concepts

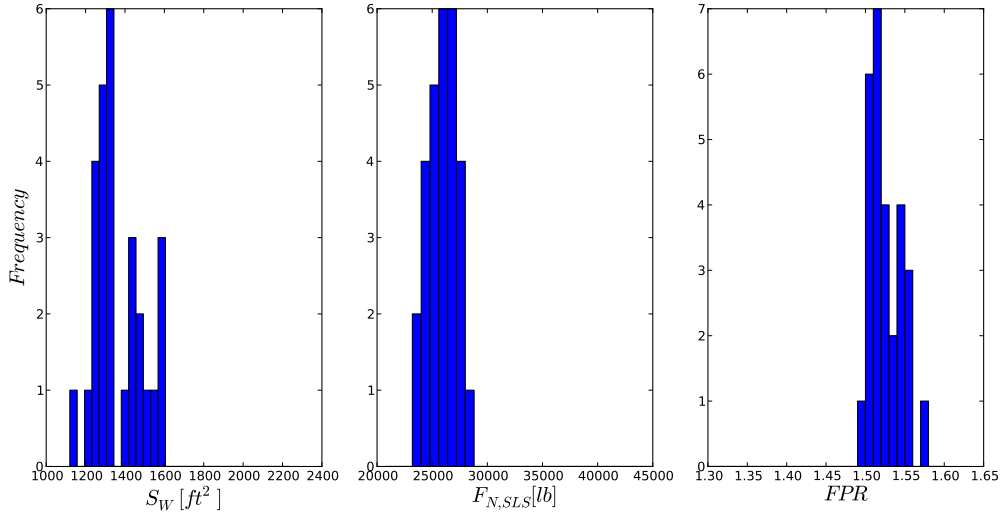


Figure 106: Design Variable Distributions

the bottom right portion of Figure 105. The two charts on the top row of Figure 105 show the proposed Pareto Distance measure that can be used to visualize complex multivariate relationships in two dimensions. Pareto Distance is plotted as a function of the design variable FPR. The points highlighted in the noise/ramp weight plot are those same points with the minimum Pareto Distance. Also evident from the figure is the region corresponding to where geared engines are most similar to the best performing direct-drive architectures. This occurs around fan pressure ratio equal to 1.55.

The method has placed geared-drive samples near the Pareto frontier of the direct-drive engines. The trends in design variables can be observed from the cluster of designs in this area. The histograms of the three continuous variables are shown in Figure 106. The particular group of samples was chosen by visual inspection of both Pareto Distance charts and objective trade space. It can be seen these designs have FPR between 1.5 and 1.55 and values of wing area and thrust on the low end of acceptable ranges. The OPR for these engines tend towards high compression, and there is no observable trend in work split. This emphasizes the benefits of PFI-based evaluation in that these designs are located in the design space where there is a high probability of being infeasible.

6.4.5 Benchmarking

One of the main goals of this research stated in Section 1.5.2 is to focus on those areas of the multiobjective design space that are most important for making concept selection decisions. Designers would like to understand the risks associated with selecting one concept alternative over another as well as the implications of preferences placed on the objectives. While visual inspection of the sample space presented in prior sections indicates clustering of designs where direct-drive engines are most similar in performance to optimal geared-drive fans, the purpose of this section is

to benchmark quantitatively the sampling performance. In others words, results will be presented here that justify the overhead computational expense of the PFI-based sampling.

First, the Pareto optimal geared turbofan that is closest in performance to the Pareto frontier of direct-drive technology is identified from the Pareto Distance chart presented above. While direct-drive fans are completely dominated, this design represents a likely point of departure from traditional architecture and pursuance of advanced technology for UHB engines. It is important to recognize that other geared designs may be more preferable given realistic objective weightings, but the shift in optimality from one concept to another is most likely in this area. As expected, this design with the lowest Pareto Distance occurs in the geared design with highest noise and low ramp weight. The engine is a geared turbofan with fixed nozzle and high OPR.

It was hypothesized that PFI-based sampling could reduce uncertainty in areas where optimality shifts between concepts. While no clearly definable shift occurs, the region surrounding the above described engine is desirable from the point of view of the decision maker. Uncertainty in this area was compared for three sampling methods: 1) PIC, 2) MOPI, and 3) MPPI. Multiobjective Probability of Improvement represents a concept evaluation method where analysis is performed sequentially and independently. Sampling based on the infill criterion MPPI is a simultaneous multiobjective optimization approach where individual Pareto frontiers of each concept are never formed. Forty adaptive function calls were used to train Kriging models of the four objectives. A Monte Carlo was run on the surrogate models in a neighborhood of design variables about the above-described engine. The neighborhood was defined as a $\pm 2.5\%$ deviation from the baseline design. Kriging predictor uncertainty was tracked for 200 random evaluations of the surrogates and averaged for each experiment. The ranges on the design variables are given in Table 22.

Table 22: Error Comparison Ranges for Monte Carlo Simulation

	Min	Baseline	Max
FPR	1.4703	1.5080	1.5457
Wing Area [ft^2]	1239	1271	1303
SLS Thrust [lb]	26013	26681	27348

The mean error is shown for the three methods in Figure 107. Seen in the figure is the significant reduction in predicted error about the specified area for the proposed sampling criterion PIC. The reason for the reduction in error is explained by the sample location for each approach. This is presented in Figure 108. The space-filling set was uniform across the three methods and is not shown for clarity. PIC, shown in Figure 108a is the only criterion to place numerous designs in areas of low ramp weight high noise for the geared architecture. This clustering of successful designs leads to the reduced uncertainty in this area. MOPI expectedly places samples evenly across the Pareto frontiers of individual concepts but has more difficulty with the direct-drive engine (Figure 108b). In searching for points that satisfy the constraints, the sampling places points off the direct-drive Pareto frontier. As a sequential optimization method, there is no knowledge of competing concept performance and evenly-spaced individual frontiers are obtained. PIC on the other hand, has omitted a large portion of the dominated designs of the geared engine (blue dots). S-Pareto dominance is taken into consideration for MPPI sampling criterion shown in Figure 108c. The effect of maximizing the probability of dominance is apparent in the direct-drive evaluations as clusters of samples can be seen at the two extremes of the Pareto frontier where improvement is most likely. A large portion of the geared design space is ignored however, as cases with low noise are more likely to be dominant over direct-drive than the opposite end of the Pareto frontier.

The PIC sampling method has found a balance between the two evaluation methodologies. On one hand, feasible designs were found in an area that MPPI avoided (high noise) but still Pareto optimal. It also improves over the MOPI criterion by evaluating

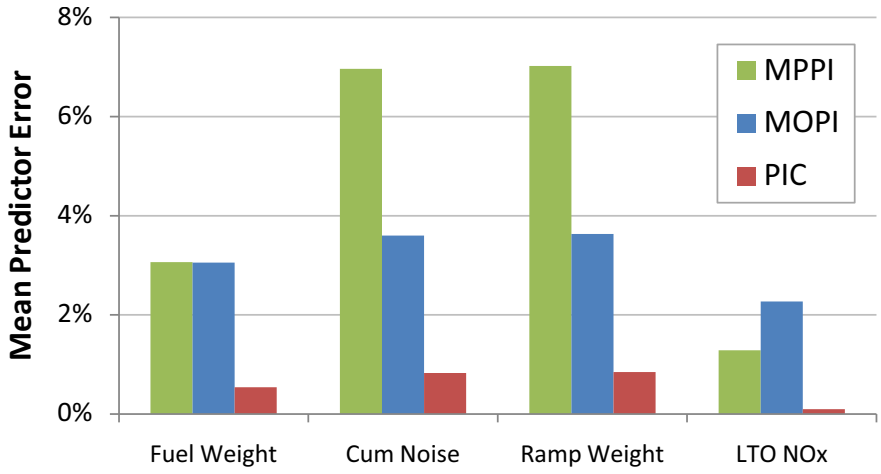
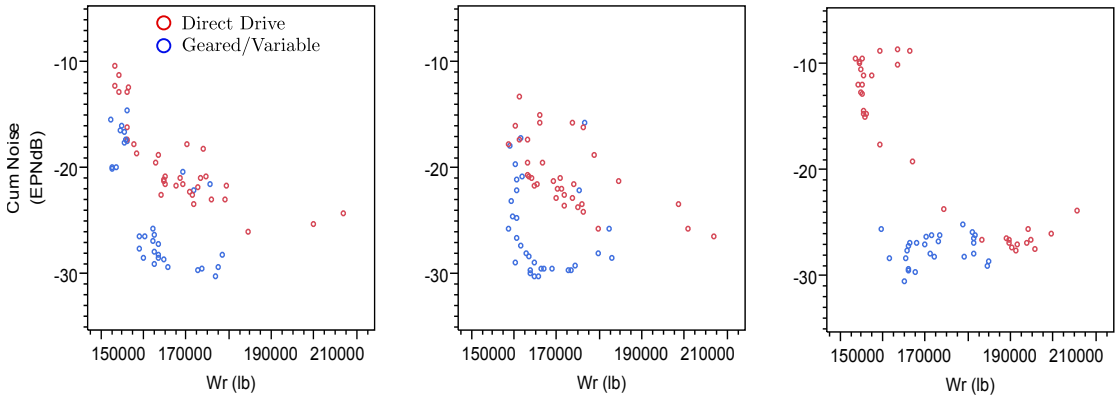


Figure 107: Predictor Error for Three Evaluation Methods



(a) PIC (b) MOPI (c) MPPI

Figure 108: Sampling for Competing Evaluation Methods

dominated concepts in areas most likely to be an improvement over the multi-concept set. Without knowing the relationship between concepts before performing the evaluation, the PFI-based approach offers a robust sampling method for models where minimizing function calls is the goal.

6.5 Summary

The PFI-based evaluation methodology concludes with data visualization and analysis presented in the preceding section. At this stage in the design process, decision makers have a more precise depiction of UHB engine performance of both geared and direct-drive technology from analytical evaluation of concept models. In performing

a quantitative assessment of concept performance with particular attention to optimality across multiple concepts as well as Pareto frontier intersections, designers can fully understand the implication of objective preferences. The designer is also provided with a set of surrogate models for each objective, constraint and concept with which to make further predictions.

The UHB design problem represents a very challenging proof-of-concept for the proposed methodology. The design space is highly constrained by the seven mission requirements, and many combinations of inputs are invalid causing a failure in one of the six coupled multidisciplinary analysis tools. The potential for models to continuously evolve throughout conceptual design places a high degree of importance on efficient model executions. In searching for Pareto optimality, designers would like to avoid spending computational resources on unusable designs and get the most information from the designs that are successful.

Another aspect of the UHB design problem is the complex relationship between the particular concepts of interest. Direct-drive engines are never preferred while there is no obvious shift from fixed to variables geometry bypass nozzles. In short, the phenomenon the methodology was set to discover did not exist. This was not known *a priori* however, so evaluation efforts were not wasted. Designs were evaluated where shifts were most probable providing still valuable insight into the design problem. While the canonical examples provided a preliminary study of sampling behavior when relationships are not ideal, the sampling performance was slightly degraded for a real engineering example. A number of model executions were performed in suboptimal areas owing to the difficulty of surrogate prediction and infill criterion formulation. The proposed methodology was nevertheless able to reduce uncertainty in regions where preference shifts are most likely, the ultimate goal of Pareto Frontier Intersection-based Evaluation.

CHAPTER VII

CONCLUSIONS

A summary of the thesis work is presented in this chapter. The scientific method continues by revisiting the research questions and addressing the hypotheses based on results of the proof-of-concept design problem and computer experiments. A summary of the contribution to the field of complex systems design will be presented followed by some recommendations and limitations observed by the proposed methodology. The thesis will conclude with identification of potential areas for further research.

7.1 Revisiting the Research Questions and Hypotheses

The purpose of this section is to revisit the research questions and hypotheses posed throughout this thesis and address them with regards to the proof-of-concept UHB design problem.

The methodological research question posed in Chapter 1 outlined the general problem and provided motivation for this research. In doing quantitative analysis for the concept evaluation and selection phases of early design, a number of issues were identified. First is the desire to reduce the number of function calls for a constantly evolving and complex model. The method must also be capable of handling constraints and failed cases while minimizing an arbitrary number of competing objectives.

It was hypothesized that a robust sampling methodology devoted to finding the Pareto frontier intersections could address all of these needs. This approach represents a new paradigm in concept evaluation, so it was further defined what it meant to be near the Pareto frontier intersection and why it is important to the decision maker. In searching for feasible designs that are 1) globally optimal across concepts and 2)

similar in performance to Pareto optimal designs of competing concepts, the decision maker can quickly identify where optimality shifts from one concept alternative to another and identify where selection decisions are most sensitive to preferences placed on the objectives.

The methodological research question led to a set of low-level research questions posed in Section 5.1.1 that dealt with specific aspects of the implementation of the PFI-based evaluation methodology. They are restated here and addressed individually based on the results obtained from implementation of the methodology on the UHB design problem.

Research Question 3.1: Does the PIC sampling criterion concentrate designs near Pareto frontier intersections of competing concepts?

Sequential sampling of competing UHB engine alternatives through PIC maximization provided distinct clustering of feasible designs along Pareto frontiers of individual concepts. The lack of a true Pareto frontier intersections made placement of subsequent designs difficult, but uncertainty was reduced in the area most likely to contain an optimality shift. The number of infeasible and failed cases was reduced with the robust sampling methodology. The proposed Pareto Distance charts combined with Pareto filtering and multivariate scatterplots provide a way to empirically locate in two dimensions the similar performing optimal designs across concepts.

Research Question 3.2: How does the PFI-based methodology compare to traditional methods of sequential and simultaneous optimization?

The difference in sampling performance was investigated by comparing the two traditional evaluation techniques. For a true comparison, the adaptive samples obtained from each method were compared using identical sets of space-filling samples. While the order of magnitude for the error is low for all methods, it is important to note that each method was surrogate-based adaptive sampling where the cost to locate sequential designs was not negligible. Therefore, if computational expense is

invested in optimizing an infill criterion at every iteration, each function call should go towards minimizing uncertainty in regions where design decisions are most costly.

Research Question 3.3: How robust is the proposed method when solving less idealized problems?

Mathematically, the proposed method is capable of handling problems with an arbitrary number of objectives and design variables. If the concept models satisfy the assumptions of Kriging (smooth, deterministic with correlated error), the probability of improvement can be estimated from Monte Carlo simulation and Normalized Pareto Distance obtained from the observed data and surrogate predictions. Additional considerations for failed cases and inequality constraints were incorporated into the methodology to account for common characteristics of real engineering problems. The method was also demonstrated to be capable of handling any number of discrete design variables by treating a unique combination as merely a new concept.

Difficulty arose as a result of the relationship between concepts. Complete dominance by the geared architectures caused many samples to be evaluated in regions of suboptimality. Samples appeared to alternate between global optimality and similar performance among concepts. In other words, the two components of the infill criterion could not be maximized concurrently.

Research Question 3.4: What is the sensitivity of PFI-based evaluation to the assumptions aimed at increasing efficiency?

The results of the canonical problems were largely used to answer this research question and establish some basic guidelines to be implemented for more costly evaluations. The $10k$ rule for an initial space-filling set of observations was adjusted slightly for likelihood of unusable (failed) cases. Knowledge of failure rate led to a 20% increase in initial population. Another assumption that was used in the evaluation was that calculation of Kriging hyperparameters occur only upon the first iteration. This assumption saved considerable computation time and still provided

surrogate accuracy in desirable areas as evidenced in Figure 107.

7.2 Summary of Contributions

The research presented in this thesis demonstrates a new approach for incorporating expensive analysis codes into the concept evaluation phase of early design. The proposed Pareto Frontier Intersection-based evaluation methodology is a novel way to address the need for a robust sampling for multiple competing concepts backed by independent models. The methodology shows demonstrated improvement over the two classes of traditional quantitative concept evaluation by reducing the number of failed cases, infeasible points and suboptimal cases as well as reducing the uncertainty in areas of design space that are most important for concept selection decisions.

Individual function calls are placed at a premium due to not only the computational expense of one successful case, but the desire to minimize overall evaluation time of multiple competing concepts. As design requirements or modeling assumptions change during the early phases of design, a high priority is placed on rapid turnaround for concept comparisons. An important point mentioned in Chapter 1 is reiterated here: too much time searching for accurate global Pareto frontiers could be wasted computation if a single technology assumption or design variable changes. This is the primary motivation behind a PFI-based evaluation methodology which is capable of reducing the number of failed/infeasible cases and focusing specifically on areas most informative to the decision maker. Through adaptive sampling of the design space, models can be executed relative to competing concepts' performance across multiple objectives and constraints.

In developing a complete methodology, a new infill criterion for Bayesian adaptive sampling was developed aimed at locating the intersection of Pareto frontiers. Pareto Intersection Closeness combined elements of traditional multiobjective optimization

with a measure of global optimality as well as proximity to competing concept designs. The extension to classical multiobjective probability of improvement represents an additional contribution of this work and builds on the most recent simultaneous evaluation techniques found in the literature.

The final contribution of this work is in the area of data visualization. The proposed Pareto Distance chart concept is a novel way of observing complex relationships in multivariate data where Pareto optimality is often difficult to infer from two dimensional projections. Pareto distance plotted as a function objectives and design variables offers a clear indication of where optimality shifts between concepts and where in the design space of each concept that shift occurs.

7.3 Recommendations

It is not expected the proposed approach for concept evaluation is appropriate for every type of problem. The UHB design problem studied in this research contained many characteristics that motivate the need for a robust search for Pareto intersections of competing concepts. This section will present some recommendations for the general methodology and identify qualities of certain problems for which PFI-based evaluation may not be appropriate. The section concludes with a modification to the proposed sampling criterion that generalizes the methodology for other types of design problems.

7.3.1 Complexity Issues

The selection of a particular optimization technique is often highly dependent on time required for a single function call. As potential for infeasibility and failed cases increases, computational expense is increased even further. A strong motivator for moving away from evolutionary algorithms is the often inordinate number of function calls required to obtain Pareto optimal designs. While EAs are desirable in that objective preferences are specified *a posteriori*, robust sampling is difficult to achieve

leading to many failed cases beyond the already expensive evaluation. For a simple model, this is less of a concern. In the time it takes to build surrogate models for each objective and maximize the infill criterion at every iteration, a large population of random individuals could be evaluated. If infeasible regions or failed cases are not likely, this further decreases the need for the proposed methodology. For sufficiently fast model executions, a comparison of competing concepts can be achieved using sequential optimization of independent concepts.

On the opposite end of complexity spectrum, PFI-based evaluation may be undesirable for problems with large number of constraints, objectives and design variables. The cost to optimize PIC is linearly increasing with constraints and objectives, as a new surrogate must be trained for each of these model outputs. Furthermore, the Kriging training cost increases exponentially with number of training points due to correlation matrix inversion. It has been shown widely in the literature that training cost is prohibitively expensive with as few as 500 training points. While Kriging was demonstrated to perform well for the problem discussed here, the methodology does not depend on this type of surrogate model. Rather, PFI-based evaluation requires merely a prediction of uncertainty along with an expected value. Substitution of surrogate model does not weaken or change the evaluation process in any way. The literature is quite deep on the subject of Bayesian models, and more sophisticated modeling techniques may be required for problems whose training costs dominate the overall evaluation.

7.3.2 Infill Criterion Weighting

For some optimization problems, designers may be more interested in obtaining a global frontier rather than focusing on the intersection of Pareto frontiers. Note that the goal for concept evaluation was to search for two things: global optimality and similar performing designs. Designs with both of these properties were said to be on

the intersection of Pareto frontiers of competing concepts. The infill criterion developed in this thesis is an aggregate objective function which balances these objectives equally. Similar to weighted decision matrices, these two objectives can be scaled according to designer preferences. The addition of weights gives more flexibility in tuning the adaptive search to the specific needs of the problem.

7.4 Suggestions for Further Research

The ideas presented in this thesis lead to many opportunities that remain unexplored. The research presented a general framework for performing concept evaluation based on Pareto frontier intersections. One specific sampling criterion was developed and demonstrated to locate the intersections among multiple competing concepts. Future sampling methods may exist that improve upon the proposed criterion or perhaps do not rely on Bayesian adaptive sampling. The specific problem provided a strong motivation for the functional form of infill criterion where other problem may warrant creation of entirely new methods.

Another aspect to the research that was only briefly explored and beyond the scope of this thesis arises from the discipline of computer science. While numerical optimization and algorithm automation is highly coupled to the computational framework and particular language in which they are implemented, the methodology was formulated independent of any computational hardware or software requirements. Two opportunities for stronger coupling between numerical optimization and computer science are readily apparent: 1) numerical precision and 2) parallelization. Precision issues were briefly mentioned in calculation of the probability of improvement. When surrogate uncertainty reduced to machine precision, designs became randomly placed. Improvement in this area could increase the accuracy of improvement predictions. The second opportunity is in the area parallel processing. There is potential for significant performance improvements if models can be run in parallel. The methodology makes no

assumptions about serial or parallel execution, and this remains largely dependent on the computational framework and model organization.

APPENDIX A

PFI-BASED EVALUATION SOURCE CODE

A.1 Kriging Generator

Listing A.1: kriging_surrogate.py

```
1  from math import log, e, sqrt
2  # pylint: disable-msg=E0611,F0401
3  from numpy import array, zeros, dot, ones, arange, eye, abs,
       vstack, exp, diag
4  from numpy.linalg import det, linalg, lstsq
5  from scipy.linalg import cho_factor, cho_solve
6  from scipy.optimize import fmin
7  from enthought.traits.api import HasTraits
8  from openmdao.lib.datatypes.api import implements
9  from openmdao.lib.drivers.api import Genetic
10 from openmdao.main.interfaces import ISurrogate
11 from openmdao.main.uncertain_distributions import
       NormalDistribution
12 from openmdao.lib.datatypes.api import Array
13
14 class KrigingSurrogate(HasTraits):
15     implements(ISurrogate)
16
17     def __init__(self,X=None,Y=None):
18         # must call HasTraits init to set up Traits stuff
19         super(KrigingSurrogate, self).__init__()
20         self.m = None #number of independent
21         self.n = None #number of training points
22         self.thetas = None
23         self.nugget = 0 #nugget smoothing parameter from [
               Sasena, 2002]
24         self.R = None
25         self.R_fact = None
26         self.mu = None
27         self.sig2 = None
28         self.log_likelihood = None
29         self.X = X
30         self.Y = Y
31         if X is not None and Y is not None:
```

```
32              self.train(X,Y)
33
34      def get_uncertain_value(self,value):
35          """Returns a NormalDistribution centered around the
                value, with a
36          standard deviation of 0."""
37          return NormalDistribution(value,0.)
38
39      def predict(self,new_x):
40          """Calculates a predicted value of the response based
                on the current
41          trained model for the supplied list of inputs.
42          """
43          if self.m == None: #untrained surrogate
44              raise RuntimeError("KrigingSurrogate has not been
                    trained, so no "
45                                 "prediction can be made")
46          r = zeros(self.n)
47          X, Y = self.X, self.Y
48          thetas = 10.**self.thetas
49          XX = array(X)
50          for i in range(self.n):
51              r[i] = sum(thetas*(XX[i]-new_x)**2.)
52          r = exp(-r)
53
54          one = ones(self.n)
55          if self.R_fact is not None:
56              rhs = vstack([(Y-dot(one, self.mu)), r, one]).T
57              R_fact = (self.R_fact[0].T,not self.R_fact[1])
58              cho = cho_solve(R_fact, rhs).T
59
60              f = self.mu + dot(r, cho[0])
61              term1 = dot(r, cho[1])
62              term2 = (1.0 - dot(one, cho[1]))**2./dot(one, cho
                    [2])
63
64          else:
65              #------LSTSQ----------
66              rhs = vstack([(Y-dot(one, self.mu)), r, one]).T
67              lsq = lstsq(self.R.T, rhs)[0].T
68
69              f = self.mu + dot(r, lsq[0])
70              term1 = dot(r, lsq[1])
71              term2 = (1.0 - dot(one, lsq[1]))**2./dot(one, lsq
                    [2])
```

```
72
73          MSE = self.sig2*(1.0-term1+term2)
74          RMSE = sqrt(abs(MSE))
75
76          return NormalDistribution(f, RMSE)
77
78      def train(self,X,Y):
79          """Train the surrogate model with the given set of
                inputs and outputs."""
80          self.X = X
81          self.Y = Y
82          self.m = len(X[0])
83          self.n = len(X)
84          thetas = zeros(self.m)
85          def _calcll(thetas):
86              self.thetas = thetas
87              self._calculate_log_likelihood()
88              return -self.log_likelihood
89          if self.thetas == None:
90              self.thetas = fmin(_calcll, thetas, disp=False,
                    ftol = 0.0001)
91          #print self.thetas
92          self._calculate_log_likelihood()
93
94      def _calculate_log_likelihood(self):
95          R = zeros((self.n, self.n))
96          X,Y = array(self.X), array(self.Y)
97          thetas = 10.**self.thetas
98          for i in range(self.n):
99              for j in arange(i+1,self.n):
100                 R[i,j] = (1-self.nugget)*e**(-sum(thetas*(X[i
                        ]-X[j])**2.)) #weighted distance formula
101         R = R + R.T + eye(self.n)
102         self.R = R
103         one = ones(self.n)
104         try:
105             self.R_fact = cho_factor(R)
106             rhs = vstack([Y, one]).T
107             R_fact = (self.R_fact[0].T,not self.R_fact[1])
108             cho = cho_solve(R_fact, rhs).T
109
110             self.mu = dot(one,cho[0])/dot(one,cho[1])
111             self.sig2 = dot(Y-dot(one,self.mu),cho_solve(self
                    .R_fact,(Y-dot(one,self.mu))))/self.n
```

```
112             self.log_likelihood = -self.n/2.*log(self.sig2)
                    -1./2.*log(abs(det(self.R)+1.e-16))
113         except (linalg.LinAlgError, ValueError):
114             #------LSTSQ----------
115             self.R_fact = None #reset this to none, so we
                    know not to use cholesky
116             rhs = vstack([Y, one]).T
117             lsq = lstsq(self.R.T,rhs)[0].T
118             self.mu = dot(one,lsq[0])/dot(one,lsq[1])
119             self.sig2 = dot(Y-dot(one,self.mu),lstsq(self.R,Y
                    -dot(one,self.mu))[0])/self.n
120             self.log_likelihood = -self.n/2.*log(self.sig2)
                    -1./2.*log(abs(det(self.R)+1.e-16))
```

A.2 Multiobjective Probability of Improvement

A.2.1 Two Objectives

Listing A.2: prob_improvement_multiobj.py

```
1  """Expected Improvement calculation for two objectives"""
2  from time import time
3  from numpy import exp, abs, pi, array,isnan
4  from scipy.special import erf
5
6  from openmdao.lib.datatypes.api import Instance, Str, ListStr
       , Enum, \
7         Float, Array,Event
8
9  from openmdao.main.component import Component
10
11 from openmdao.main.interfaces import ICaseIterator
12 from openmdao.main.uncertain_distributions import
       NormalDistribution
13
14 class MultiObjProbImprovement(Component):
15     best_cases = Instance(ICaseIterator, iotype="in",
16                           desc="CaseIterator which contains only
                                  Pareto optimal cases \
17 ____________________according to criteria")
18
19     select_type = Enum("aug",values=["aug","dom"], iotype="in
           ",
```

```
20                                                    desc="determines␣
                                                        if␣EI␣searches
                                                        ␣for␣points␣
                                                        that␣augment\
21 ␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣or␣dominate␣the␣current␣Pareto␣set")
22
23     criteria = Array(iotype="in",
24                      desc="Names␣of␣responses␣to␣maximize␣
                           expected␣improvement␣around.␣\
25 ␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣Must␣be␣NormalDistribution␣type.")
26
27     predicted_values = Array(iotype="in",dtype=
           NormalDistribution,
28                              desc="CaseIterator␣which␣contains␣
                                   NormalDistributions␣for␣each␣\
29 ␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣response␣at␣a␣location␣where␣you␣wish
       ␣to␣calculate␣EI.")
30
31     PI = Float(0.0, iotype="out", desc="The␣probability␣of␣
           improvement␣of␣the␣next␣case")
32
33     reset_y_star = Event()
34
35     def __init__(self, *args, **kwargs):
36         super(MultiObjProbImprovement, self).__init__(*args,
               **kwargs)
37         self.y_star = None
38
39     def _reset_y_star_fired(self):
40         self.y_star = None
41
42     def get_y_star(self):
43         criteria_count = len(self.criteria)
44
45         flat_crit= self.criteria.ravel()
46
47         #y_star is a 2D list of pareto points
48         y_star = []
49
50         for case in self.best_cases:
51             c = []
52             for crit in self.criteria:
53                 c.extend([o[2] for o in case.outputs if crit
                       in o[0]])
```

```
54                  #c = [o[2] for o in case.outputs if o[0] in
                        flat_crit]
55
56              if len(c) == criteria_count :
57                  y_star.append(c)
58          if not y_star: #empty y_star set means no cases met
                the criteria!
59              self.raise_exception('no cases in the provided
                    case_set had output '
60                  'matching the provided criteria , %s'%self.
                        criteria , ValueError)
61
62          #sort list on first objective
63          y_star = array(y_star)[array([i[0] for i in y_star]).
                argsort()]
64          return y_star
65
66      def _multiPI(self ,mu,sigma):
67          """Calculates the multi-objective probability of
                improvement
68          for a new point with two responses. Takes as input a
69          pareto frontier, mean and sigma of new point"""
70
71          y_star = self.y_star
72
73          PI1 = (0.5+0.5*erf((1/(2**0.5))*((y_star[0][0]-mu[0])
                /sigma[0])))
74          PI3 = (1-(0.5+0.5*erf((1/(2**0.5))*((y_star[-1][0]-mu
                [0])/sigma[0]))))\
75          *(0.5+0.5*erf((1/(2**0.5))*((y_star[-1][1]-mu[1])/
                sigma[1])))
76
77          PI2 = 0
78          if len(y_star)>1:
79              if self.select_type =="aug":
80                  for i in range(len(y_star)-1):
81                      PI2=PI2+((0.5+0.5*erf((1/(2**0.5))*((
                            y_star[i+1][0]-mu[0])/sigma[0])))\
82                      -(0.5+0.5*erf((1/(2**0.5))*((y_star[i
                            ][0]-mu[0])/sigma[0]))))\
83                       *(0.5+0.5*erf((1/(2**0.5))*((y_star[i
                            ][1]-mu[1])/sigma[1]))) #use if
                            augmenting solutions are required
84              if self.select_type == "dom":
85                  for i in range(len(y_star)-1):
```

```
86                          PI2=PI2+((0.5+0.5*erf((1/(2**0.5))*((
                                y_star[i+1][0]-mu[0])/sigma[0])))\
87                          -(0.5+0.5*erf((1/(2**0.5))*((y_star[i
                                ][0]-mu[0])/sigma[0]))))\
88                          *(0.5+0.5*erf((1/(2**0.5))*((y_star[i
                                +1][1]-mu[1])/sigma[1]))) #use if
                                dominating solutions are required
89              mcpi = PI1+PI2+PI3
90              return mcpi
91
92          def execute(self):
93              """ Calculates the expected improvement of
94              the model at a given point.
95              """
96              mu = [objective.mu for objective in self.
                    predicted_values]
97              sig = [objective.sigma for objective in self.
                    predicted_values]
98              if self.y_star == None:
99                  self.y_star = self.get_y_star()
100             self.PI = self._multiPI(mu,sig)
```

A.2.2 Three or More Objectives

Listing A.3: pi_MCS.py

```
1  """Expected Improvement calculation for three or more
       objectives"""
2  from time import time
3  from numpy import exp, abs, pi, array,isnan, diag
4  from scipy.special import erf
5  from scipy import random
6
7  from openmdao.lib.datatypes.api import Instance, Str, ListStr
       , Enum, \
8          Float, Array,Event, Int
9  from openmdao.lib.components.api import ParetoFilter
10 from openmdao.main.component import Component
11 from openmdao.main.api import Case
12 from openmdao.main.interfaces import ICaseIterator
13 from openmdao.main.uncertain_distributions import
       NormalDistribution
14 from openmdao.lib.caseiterators.api import ListCaseIterator
15
16 class PI_MCS(Component):
17     best_cases = Instance(ICaseIterator, iotype="in",
```

```
18                      desc="CaseIterator which contains only
                            Pareto optimal cases \
19 ____________________according to criteria")
20
21     criteria = Array(iotype="in",
22                      desc="Names of responses to maximize PI
                            around. \
23 ____________________Must be NormalDistribution type.")
24
25     predicted_values = Array(iotype="in",dtype=
           NormalDistribution,
26                              desc="CaseIterator which contains
                                   NormalDistributions for each \
27 ________________________response at a location where you wish
       to calculate EI.")
28     n = Int(iotype='in',desc='number of monte carlo samples
           to calc probability')
29
30     PI = Float(0.0, iotype="out", desc="The probability of
           improvement of the next case")
31
32     reset_y_star = Event()
33
34     def __init__(self, *args, **kwargs):
35         super(PI_MCS, self).__init__(*args, **kwargs)
36         self.y_star = None
37
38     def _reset_y_star_fired(self):
39         self.y_star = None
40
41     def get_y_star(self):
42         criteria_count = len(self.criteria)
43         flat_crit= self.criteria.ravel()
44         y_star = [] #y_star is a 2D list of pareto points
45         for case in self.best_cases:
46             c = []
47             for crit in self.criteria:
48                 c.extend([o[2] for o in case.outputs if crit
                        in o[0]])
49                 #c = [o[2] for o in case.outputs if o[0] in
                        flat_crit]
50
51             if len(c) == criteria_count :
52                 y_star.append(c)
```

```
53          if not y_star: #empty y_star set means no cases met
                the criteria!
54              self.raise_exception('no cases in the provided
                    case_set had output '
55                  'matching the provided criteria , %s'%self.
                        criteria , ValueError)
56          y_star = array(y_star)[array([i[0] for i in y_star]).
                argsort()] #sort list on first objective
57          return y_star
58
59      def _dom(self ,a,b):
60          """determines if a completely dominates b
61         returns True is if does
62      """
63          comp = [c1<c2 for c1,c2 in zip(a,b)]
64          if sum(comp)==len(self.criteria):
65              return True
66          return False
67
68      def _multiPI(self ,mu,sigma):
69          cov = diag(array(sigma)**2)
70          rands = random.multivariate_normal(mu,cov,self.n)
71          num = 0 #number of cases that dominate the current
                Pareto set
72          for random_sample in rands:
73              for par_point in self.best_cases:
74                  par_point = [p[2] for p in par_point.outputs]
75                  if self._dom(par_point,random_sample):
76                      num = num+1
77                      break
78          pi = (self.n-num)/float(self.n)
79          return pi
80
81      def execute(self):
82          """ Calculates the expected improvement of
83          the model at a given point.
84          """
85          mu = [objective.mu for objective in self.
                predicted_values]
86          sig = [objective.sigma for objective in self.
                predicted_values]
87          if self.y_star == None:
88              self.y_star = self.get_y_star()
89          self.PI = self._multiPI(mu,sig)
```

A.3 Normalized Pareto Distance

Listing A.4: pareto_min_dist.py

```
1  from time import time
2  from numpy import exp, abs, pi, array,isnan,sum,sqrt,argsort,
       min
3  from scipy.special import erf
4  from scipy.integrate import dblquad
5
6  from openmdao.lib.datatypes.api import Instance, Str, ListStr
       , Enum, \
7         Float, Array, Event, List
8
9  from openmdao.main.component import Component
10
11 from openmdao.main.interfaces import ICaseIterator
12 from openmdao.main.uncertain_distributions import
       NormalDistribution
13 from time import time
14
15 class Pareto_Min_Dist(Component):
16     """Computes the probability that any given point from the
           primary concept
17     will interesect the pareto frontiers of some other
           concepts.
18     """
19     primary_pareto = Instance(ICaseIterator, iotype="in",
20                               desc="List of CaseIterators containing
                                      individual concept local Pareto points
                                      ")
21
22     global_pareto = Instance(ICaseIterator, iotype="in",
23                              desc="List of CaseIterators containing
                                     global Pareto points")
24
25     criteria = ListStr(iotype="in",dtype="str",
26                              desc="Names of responses to maximize
                                     expected improvement around. "
27                                   "Must be NormalDistribution type.
                                         ")
28
29     predicted_values = Array(iotype="in",dtype=
           NormalDistribution,
```

```
30                                  desc="CaseIterator which
                                        contains a NormalDistribution
                                        "
31                                      "for each response at a
                                            location where you wish
                                            to "
32                                      "calculate EI.")
33
34      dist = Float(0.0, iotype="out",
35                   desc="minimum distance from a point to other
                        pareto set ")
36
37      reset_pareto = Event()
38
39      def __init__(self,*args,**kwargs):
40          super(Pareto_Min_Dist, self).__init__(*args, **kwargs
                )
41          self.y_star_other = None
42
43      def _reset_pareto_fired(self):
44          self.y_star_other = None
45
46      def get_pareto(self):
47          y_star_other = []
48
49          c = []
50          #find the pareto points which are in the
                global_pareto but not in the primary_pareto
51          other_pareto = [case for case in self.global_pareto
                if case not in self.primary_pareto]
52
53          #for case in self.pareto:
54          for case in other_pareto:
55              for objective in case.outputs:
56                  for crit in self.criteria:
57                      if crit in objective[0]:
58                          c.append(objective[2])
59              if c != [] :
60                  y_star_other.append(c)
61              c = []
62
63          return y_star_other
64
65      def _calc_min_dist(self,p,y_star_other):
66          """Computes the distance from a candidate point
```

```
67          to closest Pareto point of other frontier.
68          """
69          dists = []
70          for y in y_star_other:
71              d = sum([abs((A-B)/B) for A,B in zip(p,y)])
72              dists.append(d)
73          #try
74          return min(dists)
75
76      def execute(self):
77          mu = [objective.mu for objective in self.
                predicted_values]
78          #mu.reverse() #MAKE SURE THIS IS RIGHT!!!
79
80          if self.y_star_other == None:
81              self.y_star_other = self.get_pareto()
82          if self.y_star_other:
83              self.dist = self._calc_min_dist(mu, self.
                    y_star_other)
84          else:
85              self.dist = 0
```

APPENDIX B

CANONICAL PROBLEM IMPLEMENTATION

B.1 Algebraic Sample Concepts

Listing B.1: concepts.py

```
1  from openmdao.main.api import Component
2  from openmdao.lib.datatypes.api import Float
3  from numpy import pi, cos, sin
4
5  class ConceptA(Component):
6      x = Float(0,iotype="in",low=-1.0,high=1.0)
7      y = Float(0,iotype="in",low=0.,high=pi/2.)
8
9      f1 = Float(0.,iotype="out")
10     f2 = Float(0.,iotype="out")
11
12     def execute(self):
13         self.f1 = -(1-self.x**2)*cos(self.y)+2.0
14         self.f2 =  -(1-self.x**2)*sin(self.y)+1.0
15
16 class ConceptB(Component):
17     x = Float(0,iotype="in",low=-1.0,high=1.0)
18     y = Float(0,iotype="in",low=0.,high=pi/2.)
19
20     f1 = Float(0.,iotype="out")
21     f2 = Float(0.,iotype="out")
22
23     def execute(self):
24         self.f1 = -(1-self.x**2)*cos(self.y)+1.5
25         self.f2 =  -(1-self.x**2)*sin(self.y)+1.5
26
27 class ConceptC(Component):
28     x = Float(0,iotype="in",low=-1.0,high=1.0)
29     y = Float(0,iotype="in",low=0.,high=pi/2.)
30
31     f1 = Float(0.,iotype="out")
32     f2 = Float(0.,iotype="out")
33
34     def execute(self):
```

```
35          self.f1 = -(1-self.x**2)*cos(self.y)+3.0
36          self.f2 =  -(1-self.x**2)*sin(self.y)+2.0
37
38  class ConceptD(Component):
39      x = Float(0,iotype="in",low=-1.0,high=1.0)
40      y = Float(0,iotype="in",low=0.,high=pi/2.)
41      z = Float(0,iotype="in",low=0.,high=pi/2.)
42
43      f1 = Float(0.,iotype="out")
44      f2 = Float(0.,iotype="out")
45      f3 = Float(0.,iotype="out")
46
47      def execute(self):
48          self.f1 = -(1-self.x**2)*cos(self.y)+2.0
49          self.f2 =  -(1-self.x**2)*sin(self.y)*cos(self.z)+1.0
50          self.f3 =  -(1-self.x**2)*sin(self.z)*sin(self.y)+1.0
51
52  class ConceptE(Component):
53      x = Float(0,iotype="in",low=-1.0,high=1.0)
54      y = Float(0,iotype="in",low=0.,high=pi/2.)
55      z = Float(0,iotype="in",low=0.,high=pi/2.)
56
57      f1 = Float(0.,iotype="out")
58      f2 = Float(0.,iotype="out")
59      f3 = Float(0.,iotype="out")
60
61      def execute(self):
62          self.f1 = -(1-self.x**2)*cos(self.y)+1.5
63          self.f2 =  -(1-self.x**2)*sin(self.y)*cos(self.z)+1.5
64          self.f3 =  -(1-self.x**2)*sin(self.z)*sin(self.y)+1.5
65
66  class ConceptF(Component):
67      x = Float(0,iotype="in",low=-1.0,high=1.0)
68      y = Float(0,iotype="in",low=0.,high=pi/2.)
69      z = Float(0,iotype="in",low=0.,high=pi/2.)
70      w = Float(0,iotype="in",low=0.,high=pi/2.)
71
72      f1 = Float(0.,iotype="out")
73      f2 = Float(0.,iotype="out")
74      f3 = Float(0.,iotype="out")
75      f4 = Float(0.,iotype="out")
76
77      def execute(self):
78          self.f1 = -(1-self.x**2)*cos(self.y)+2.0
79          self.f2 =  -(1-self.x**2)*sin(self.y)*cos(self.z)+1.0
```

```
80          self.f3 =  -(1-self.x**2)*sin(self.z)*sin(self.y)+1.0
81          self.f4 =  -(1-self.x**2)*sin(self.z)*sin(self.y)*cos
                (self.w)+1.0
82
83
84  class ConceptG(Component):
85      x = Float(0,iotype="in",low=-1.0,high=1.0)
86      y = Float(0,iotype="in",low=0.,high=pi/2.)
87      z = Float(0,iotype="in",low=0.,high=pi/2.)
88      w = Float(0,iotype="in",low=0.,high=pi/2.)
89
90      f1 = Float(0.,iotype="out")
91      f2 = Float(0.,iotype="out")
92      f3 = Float(0.,iotype="out")
93      f4 = Float(0.,iotype="out")
94
95      def execute(self):
96          self.f1 = -(1-self.x**2)*cos(self.y)+1.5
97          self.f2 =  -(1-self.x**2)*sin(self.y)*cos(self.z)+1.5
98          self.f3 =  -(1-self.x**2)*sin(self.z)*sin(self.y)+1.5
99          self.f4 =  -(1-self.x**2)*sin(self.z)*sin(self.y)*cos
                (self.w)+1.5
```

B.2 Truss Solver

Listing B.2: concepts.py

```
1  from openmdao.main.api import Component, Case
2  from openmdao.lib.datatypes.api import Float, Instance
3  from openmdao.main.interfaces import ICaseRecorder
4
5  from numpy import pi, cos, sin, sqrt,array,vstack,hstack,
       reshape,zeros
6  from scipy import linalg
7
8  P1 = 30000
9  P2 = -30000
10 class Truss(Component):
11
12     def _kbar(self,elem,e,a):
13         """Calculates individual stiffness matrix for an
               element"""
14         x1 = self.XG[elem[0]-1]
15         x2 = self.XG[elem[1]-1]
16         y1 = self.YG[elem[0]-1]
17         y2 = self.YG[elem[1]-1]
```

```
18          z1 = self.ZG[elem[0]-1]
19          z2 = self.ZG[elem[1]-1]
20          Lbar = sqrt((x2-x1)**2+(y2-y1)**2+(z2-z1)**2) #
                element length
21          keq = e*a/Lbar #equivalent stiffness matrix
22          Cx = (x2-x1)/Lbar
23          Cy = (y2-y1)/Lbar
24          Cz = (z2-z1)/Lbar
25          l = array([[Cx**2,Cx*Cy,Cx*Cz],[Cx*Cy,Cy**2,Cy*Cz],[
                Cx*Cz,Cy*Cz,Cz**2]])
26          trans = vstack([hstack([l,-l]),hstack([-l,l])])
27          skele = keq*trans
28          vol = Lbar*a
29          return skele,vol
30
31      def _kglobal(self,skele,elem):
32          startnode = elem[0]
33          endnode = elem[1]
34          c1 = startnode+(2*(startnode-1))-1
35          c2 = endnode+(2*(endnode-1))-1
36          int = zeros([self.nnodes*3]*2)
37          int[c1:c1+3,c1:c1+3] = skele[0:3,0:3]
38          int[c1:c1+3,c2:c2+3] = skele[0:3,3:6]
39          int[c2:c2+3,c1:c1+3] = skele[3:6,0:3]
40          int[c2:c2+3,c1:c1+3] = skele[3:6,3:6]
41          return int
42
43      def _kconstrain(self,node_num,nodefix):
44          start = node_num+2*node_num
45          for i in range(3):
46              if nodefix[i] == 1:
47                  self.skglo[start+i,:] = 0
48                  self.skglo[:,start+i] = 0
49                  self.skglo[start+i,start+i] = 1
50
51      def _solve(self):
52          disps = linalg.lstsq(self.skglo,self.f)
53          disps = disps[0]
54          self.d_x = disps[0]
55          self.d_y = disps[1]
56          self.d_z = disps[2]
57          self.d_total = sqrt(self.d_x**2+self.d_y**2+self.d_z
                **2)
58
59  class TrussA(Truss):
```

```
60      a1 = Float(1.,iotype="in",low=1.,high=4.0)
61      a2 = Float(1.,iotype="in",low=1.,high=4.0)
62      a3 = Float(1.,iotype="in",low=1.,high=4.0)
63      recorder = Instance(ICaseRecorder, desc = 'Records cases'
            )
64
65      volume = Float(0.,iotype="out")
66      d_total = Float(0.,iotype="out")
67
68      def __init__(self,doc=None):
69          super(TrussA,self).__init__(doc)
70          self.nnodes = 4
71          self.nelems = 3
72          self.NODEFIX = array
                ([[0,0,1],[1,1,1],[1,1,1],[1,1,1]])
73          self.FORCE = array([[P1,P2
                ,0],[0,0,0],[0,0,0],[0,0,0]])
74          self.ELEMDEF = array([[1,2],[1,3],[1,4]])
75          self.E = 30e6*array([1,1,1])
76          self.XG = array([0,-60,0,60])
77          self.YG = array([0,120,120,120])
78          self.ZG = array([0,0,0,0])
79
80      def execute(self):
81          self.A = array([self.a1,self.a2,self.a3])
82          self.skglo = zeros([self.nnodes*3]*2)
83          self.f = self.FORCE.reshape([self.nnodes*3])
84          self.volume  = 0
85          for ELEM,E,A in zip(self.ELEMDEF,self.E,self.A):
86              skele,v = self._kbar(ELEM,E,A)
87              self.volume = self.volume+v
88              self.skglo = self.skglo+self._kglobal(skele,ELEM)
89          for i,FIX in enumerate(self.NODEFIX):
90              self._kconstrain(i,FIX)
91          self._solve()
92          case_inputs = [('a1',None,self.a1),
93                                      ('a2',None,self.a2),
94                                      ('a3',None,self.a3)]
95          case_outputs = [('d_total',None,self.d_total),
96                                          ('volume',None,self.
                                              volume)]
97          #self.recorder.record(Case(inputs=case_inputs,
                outputs=case_outputs))
98
99  class TrussB(Truss):
```

```
100     a1 = Float(1.,iotype="in",low=1.,high=4.0)
101     a2 = Float(1.,iotype="in",low=1.,high=4.0)
102     a3 = Float(1.,iotype="in",low=1.,high=4.0)
103     a4 = Float(1.,iotype="in",low=1.,high=4.0)
104     a5 = Float(1.,iotype="in",low=1.,high=4.0)
105     recorder = Instance(ICaseRecorder, desc = 'Records cases'
            )
106
107     volume = Float(0.,iotype="out")
108     d_total = Float(0.,iotype="out")
109
110     def __init__(self,doc=None):
111         super(TrussB,self).__init__(doc)
112
113         self.nnodes = 6
114         self.nelems = 5
115         self.NODEFIX = array
                ([[0,0,1],[1,1,1],[1,1,1],[1,1,1],[1,1,1],[1,1,1]])

116         self.FORCE = array([[P1,P2
                ,0],[0,0,0],[0,0,0],[0,0,0],[0,0,0],[0,0,0]])
117         self.ELEMDEF = array([[1,2],[1,3],[1,4],[1,5],[1,6]])
118         self.E = 30e6*array([1,1,1,1,1])
119         self.XG = array([0,-60,-30,0,30,60])
120         self.YG = array([0,120,120,120,120,120])
121         self.ZG = array([0,0,0,0,0,0])
122
123     def execute(self):
124         self.A = array([self.a1,self.a2,self.a3,self.a4,self.
                a5])
125         self.skglo = zeros([self.nnodes*3]*2)
126         self.f = self.FORCE.reshape([self.nnodes*3])
127         self.volume  = 0
128         for ELEM,E,A in zip(self.ELEMDEF,self.E,self.A):
129             skele,v = self._kbar(ELEM,E,A)
130             self.volume = self.volume+v
131             self.skglo = self.skglo+self._kglobal(skele,ELEM)
132         for i,FIX in enumerate(self.NODEFIX):
133             self._kconstrain(i,FIX)
134         self._solve()
135         case_inputs = [('a1',None,self.a1),
136                                    ('a2',None,self.a2),
137                                    ('a3',None,self.a3),
138                                    ('a4',None,self.a4),
139                                    ('a5',None,self.a5)]
```

```
140         case_outputs = [('d_total',None,self.d_total),
141                                         ('volume',None,self.
                                            volume)]
142         #self.recorder.record(Case(inputs=case_inputs,
                outputs=case_outputs))
143
144 class TrussC(Truss):
145     a1 = Float(1.,iotype="in",low=1.,high=4.0)
146     a2 = Float(1.,iotype="in",low=1.,high=4.0)
147     recorder = Instance(ICaseRecorder, desc = 'Records_cases'
            )
148
149     volume = Float(0.,iotype="out")
150     d_total = Float(0.,iotype="out")
151
152     def __init__(self,doc=None):
153         super(TrussC,self).__init__(doc)
154
155         self.nnodes = 3
156         self.nelems = 2
157         self.NODEFIX = array([[0,0,1],[1,1,1],[1,1,1]])
158         self.FORCE = array([[P1,P2,0],[0,0,0],[0,0,0]])
159         self.ELEMDEF = array([[1,2],[1,3]])
160         self.E = 30e6*array([1,1])
161         self.XG = array([0,-60,0])
162         self.YG = array([0,120,120])
163         self.ZG = array([0,0,0])
164
165     def execute(self):
166         self.A = array([self.a1,self.a2])
167         self.skglo = zeros([self.nnodes*3]*2)
168         self.f = self.FORCE.reshape([self.nnodes*3])
169         self.volume  = 0
170         for ELEM,E,A in zip(self.ELEMDEF,self.E,self.A):
171             skele,v = self._kbar(ELEM,E,A)
172             self.volume = self.volume+v
173             self.skglo = self.skglo+self._kglobal(skele,ELEM)
174         for i,FIX in enumerate(self.NODEFIX):
175             self._kconstrain(i,FIX)
176         self._solve()
177         case_inputs = [('a1',None,self.a1),
178                                         ('a2',None,self.a2)]
179         case_outputs = [('d_total',None,self.d_total),
180                                         ('volume',None,self.
                                            volume)]
```

```
181         #self.recorder.record(Case(inputs=case_inputs,
                outputs=case_outputs))
182
183 class TrussD(Truss):
184     a1 = Float(1.,iotype="in",low=1.,high=4.0)
185     a2 = Float(1.,iotype="in",low=1.,high=4.0)
186     recorder = Instance(ICaseRecorder, desc = 'Records_cases'
            )
187
188     volume = Float(0.,iotype="out")
189     d_total = Float(0.,iotype="out")
190
191     def __init__(self,doc=None):
192         super(TrussD,self).__init__(doc)
193
194         self.nnodes = 3
195         self.nelems = 2
196         self.NODEFIX = array([[0,0,1],[1,1,1],[1,1,1]])
197         self.FORCE = array([[P1,P2,0],[0,0,0],[0,0,0]])
198         self.ELEMDEF = array([[1,2],[1,3]])
199         self.E = 30e6*array([1,1])
200         self.XG = array([0,0,60])
201         self.YG = array([0,120,120])
202         self.ZG = array([0,0,0])
203
204     def execute(self):
205         self.A = array([self.a1,self.a2])
206         self.skglo = zeros([self.nnodes*3]*2)
207         self.f = self.FORCE.reshape([self.nnodes*3])
208         self.volume  = 0
209         for ELEM,E,A in zip(self.ELEMDEF,self.E,self.A):
210             skele,v = self._kbar(ELEM,E,A)
211             self.volume = self.volume+v
212             self.skglo = self.skglo+self._kglobal(skele,ELEM)
213         for i,FIX in enumerate(self.NODEFIX):
214             self._kconstrain(i,FIX)
215         self._solve()
216         case_inputs = [('a1',None,self.a1),
217                                    ('a2',None,self.a2)]
218         case_outputs = [('d_total',None,self.d_total),
219                                        ('volume',None,self.
                                            volume)]
220         #self.recorder.record(Case(inputs=case_inputs,
                outputs=case_outputs))
221
```

```
222 class TrussE(Truss):
223     a1 = Float(1.,iotype="in",low=1.,high=4.0)
224     a2 = Float(1.,iotype="in",low=1.,high=4.0)
225     a3 = Float(1.,iotype="in",low=1.,high=4.0)
226
227     recorder = Instance(ICaseRecorder, desc = 'Records cases'
            )
228
229     volume = Float(0.,iotype="out")
230     d_total = Float(0.,iotype="out")
231
232     def __init__(self,doc=None):
233         super(TrussE,self).__init__(doc)
234
235         self.nnodes = 4
236         self.nelems = 3
237         self.NODEFIX = array
                ([[0,0,1],[1,1,1],[1,1,1],[1,1,1]])
238         self.FORCE = array([[P1,P2
                ,0],[0,0,0],[0,0,0],[0,0,0]])
239         self.ELEMDEF = array([[1,2],[1,3],[1,4]])
240         self.E = 30e6*array([1,1,1])
241         self.XG = array([0,-60,-30,0])
242         self.YG = array([0,120,120,120])
243         self.ZG = array([0,0,0,0])
244
245     def execute(self):
246         self.A = array([self.a1,self.a2,self.a3])
247         self.skglo = zeros([self.nnodes*3]*2)
248         self.f = self.FORCE.reshape([self.nnodes*3])
249         self.volume  = 0
250         for ELEM,E,A in zip(self.ELEMDEF,self.E,self.A):
251             skele,v = self._kbar(ELEM,E,A)
252             self.volume = self.volume+v
253             self.skglo = self.skglo+self._kglobal(skele,ELEM)
254         for i,FIX in enumerate(self.NODEFIX):
255             self._kconstrain(i,FIX)
256         self._solve()
257         case_inputs = [('a1',None,self.a1),
258                                   ('a2',None,self.a2),
259                                   ('a3',None,self.a3)]
260         case_outputs = [('d_total',None,self.d_total),
261                                     ('volume',None,self.
                                        volume)]
```

```
262         #self.recorder.record(Case(inputs=case_inputs,
                outputs=case_outputs))
263
264 class TrussF(Truss):
265     a1 = Float(1.,iotype="in",low=1.,high=4.0)
266     a2 = Float(1.,iotype="in",low=1.,high=4.0)
267     a3 = Float(1.,iotype="in",low=1.,high=4.0)
268
269     recorder = Instance(ICaseRecorder, desc = 'Records_cases'
            )
270
271     volume = Float(0.,iotype="out")
272     d_total = Float(0.,iotype="out")
273
274     def __init__(self,doc=None):
275         super(TrussF,self).__init__(doc)
276
277         self.nnodes = 4
278         self.nelems = 3
279         self.NODEFIX = array
                ([[0,0,1],[1,1,1],[1,1,1],[1,1,1]])
280         self.FORCE = array([[P1,P2
                ,0],[0,0,0],[0,0,0],[0,0,0]])
281         self.ELEMDEF = array([[1,2],[1,3],[1,4]])
282         self.E = 30e6*array([1,1,1])
283         self.XG = array([0,0,30,60])
284         self.YG = array([0,120,120,120])
285         self.ZG = array([0,0,0,0])
286
287     def execute(self):
288         self.A = array([self.a1,self.a2,self.a3])
289         self.skglo = zeros([self.nnodes*3]*2)
290         self.f = self.FORCE.reshape([self.nnodes*3])
291         self.volume  = 0
292         for ELEM,E,A in zip(self.ELEMDEF,self.E,self.A):
293             skele,v = self._kbar(ELEM,E,A)
294             self.volume = self.volume+v
295             self.skglo = self.skglo+self._kglobal(skele,ELEM)
296         for i,FIX in enumerate(self.NODEFIX):
297             self._kconstrain(i,FIX)
298         self._solve()
299         case_inputs = [('a1',None,self.a1),
300                                    ('a2',None,self.a2),
301                                    ('a3',None,self.a3)]
302         case_outputs = [('d_total',None,self.d_total),
```

```
303                                                 ('volume',None,self.
                                                        volume)]
304             #self.recorder.record(Case(inputs=case_inputs,
                    outputs=case_outputs))
```

APPENDIX C

CANONICAL PROBLEM RESULTS

C.1 Zero-Overhead Methods

The following results are for those methods where the analysis cost is entirely from function calls. These differ from Bayesian adaptive sampling methods in Section 5.2.2 where the cost to locate a new sample may be greater than the cost to run many more evaluations of the model. In such a situation where function calls are extremely cheap, a sequential optimization method may be more favorable. Both DOE and NSGA-II are presented as representative sequential evaluation methods.

C.1.1 DOE Method

A Latin Hypercube DOE was run for each concept independently then Kriging surrogate fit through the results. The experiment was repeated for increasing number of total samples (up to 100) and repeated 10 times at each sampling to smooth out anomalies. An example experiment is shown in Figure 109 for 60 total function calls (maximum number of samples for adaptive sampling experiments). Figure 110 shows that the experiment is truly space-filling.

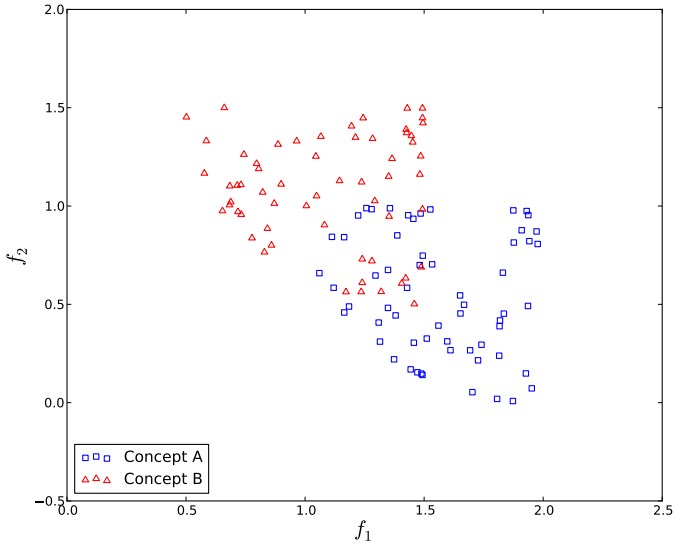
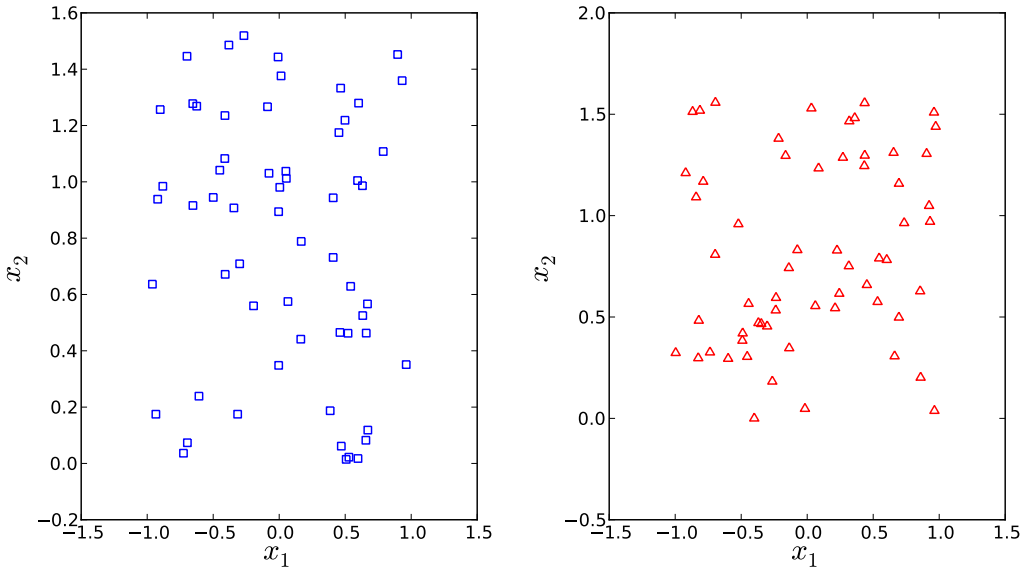


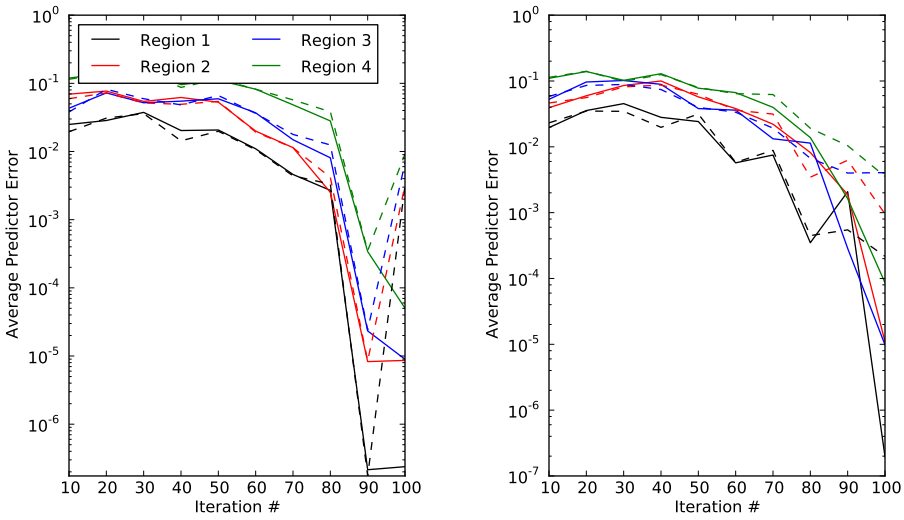
Figure 109: DOE Sampling - 60 Cases



(a) Concept A

(b) Concept B

Figure 110: Design Variable Clustering with DOE Sampling



(a) Concept A

(b) Concept B

Figure 111: Iteration History for DOE Sampling

C.1.2 NSGA-II

The NSGA-II implementation was obtained from the open-source plug-in for Python called ECsPy [12]. Figure 112 shows a sample experiment of 60 total function calls. Assuming a GA population of ten (five times number of design variables), this allowed for five iterations of crossover/mutation for six total generations including the initial set. The design variables are shown in Figure 113. Clear clustering can be seen around $x_1 = 0$ for Concept A. Though not as pronounced, the designs are trending toward this region for Concept B as well.

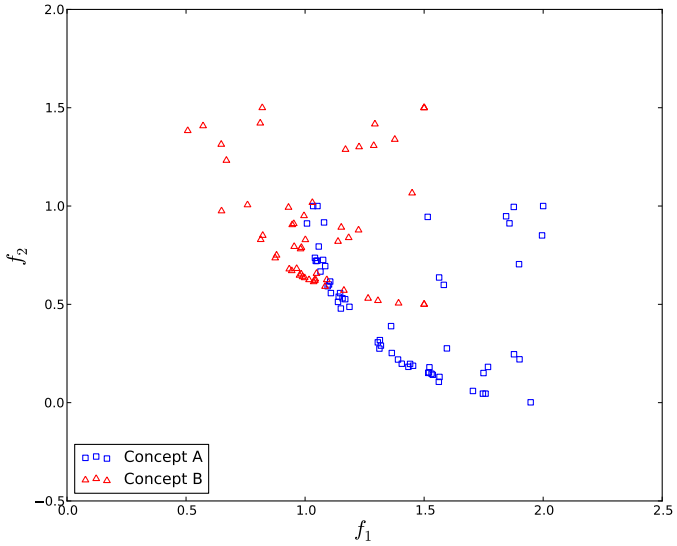
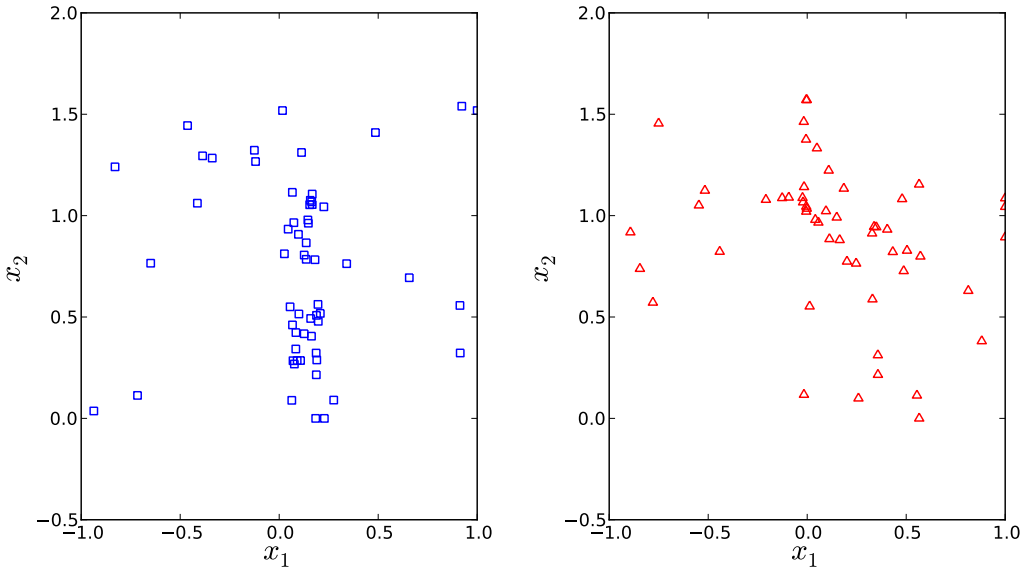


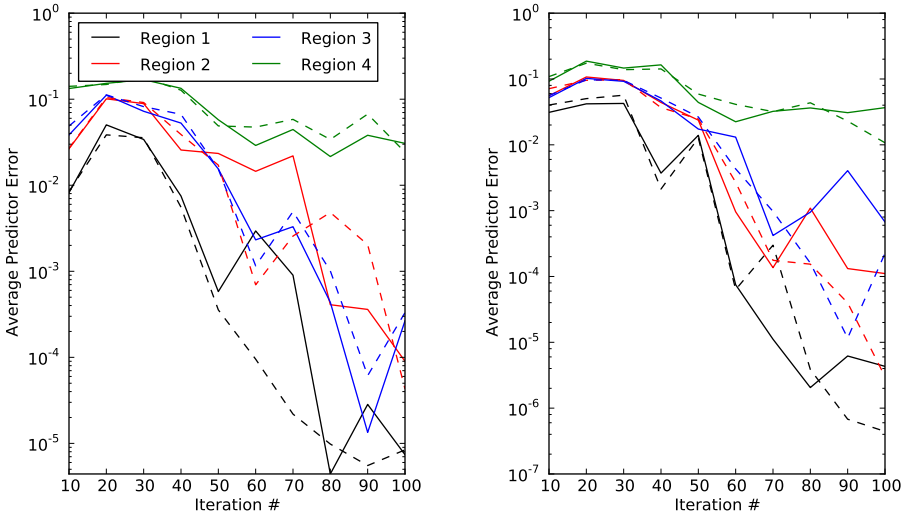
Figure 112: NSGA-II Sampling - 60 Cases



(a) Concept A

(b) Concept B

Figure 113: Design Variable Clustering with NSGA-II Sampling



(a) Concept A (b) Concept B

Figure 114: Iteration History for NSGA-II Sampling

C.2 Algebraic Sample of Four Objectives

Experiment 3.2 continues here with expansion of the algebraic sample problem into four objectives according to Equations 64-67.

$$f_1^{(k)}(x) = -(1-x_1^2)cos(x_2) + M^{(k)} \quad (64)$$

$$f_2^{(k)}(x) = -(1-x_1^2)sin(x_2)cos(x_3) + N^{(k)} \quad (65)$$

$$f_3^{(k)}(x) = -(1-x_1^2)sin(x_2)sin(x_3) + O^{(k)} \quad (66)$$

$$f_4^{(k)}(x) = -(1-x_1^2)sin(x_2)sin(x_3)cos(x_4) + P^{(k)} \quad (67)$$

$$\text{Subject to:} \qquad -1 < x_1 < 1, \quad 0 < x_{2,3,4} < \frac{\pi}{2}$$

where $[M^{(A)}, N^{(A)}, O^{(A)}, P^{(A)}]$ is $[2.0, 1.0, 1.0, 1.0]$, and $[M^{(B)}, N^{(B)}, O^{(B)}, P^{(B)}]$ is $[1.5, 1.5, 1.5, 1.5]$. The random sampling is shown Figure 115.

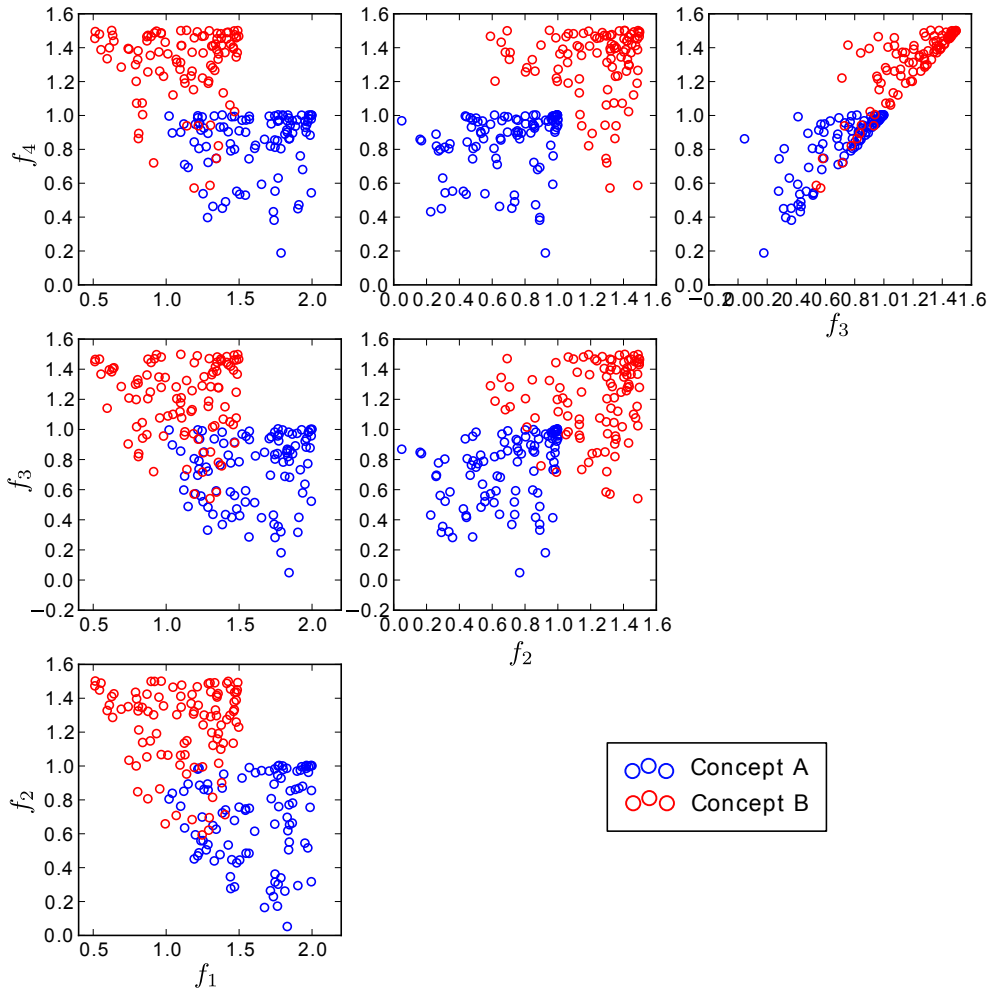


Figure 115: Monte Carlo Sampling

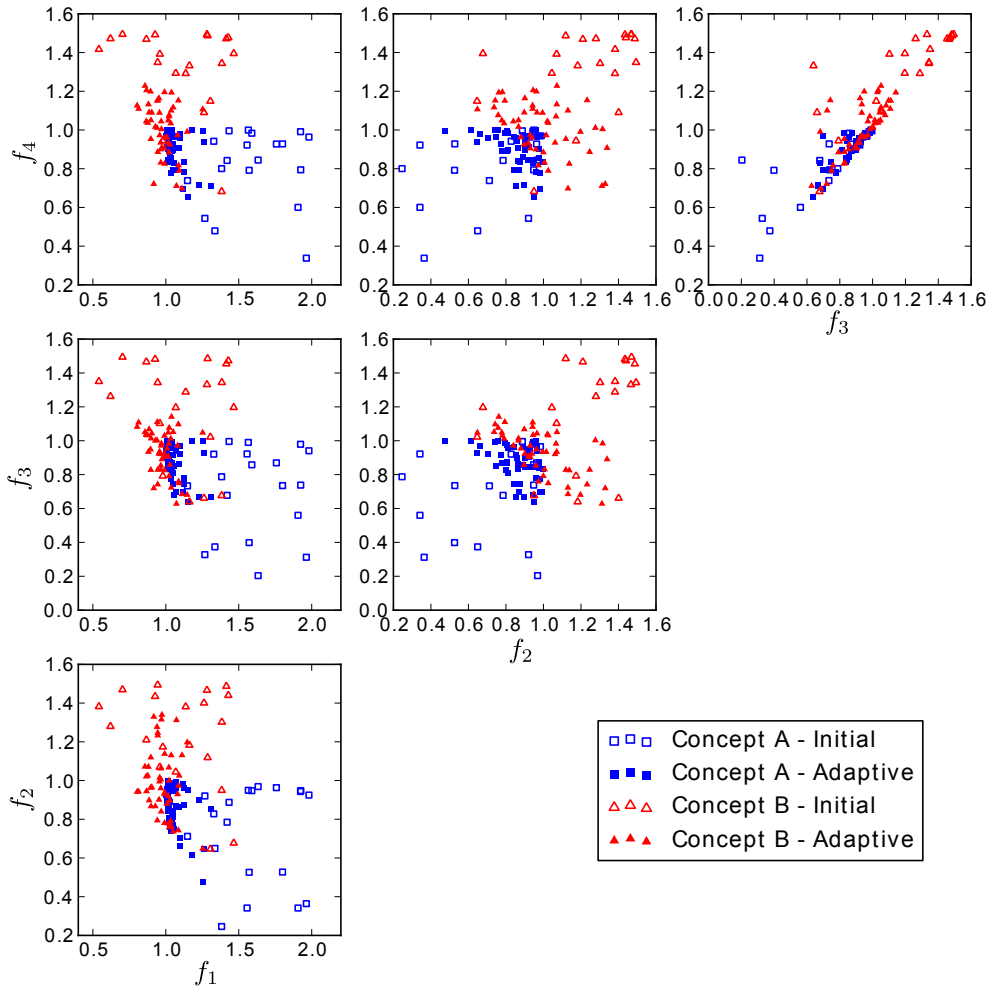


Figure 116: PFI-Based Sampling in Four Objectives

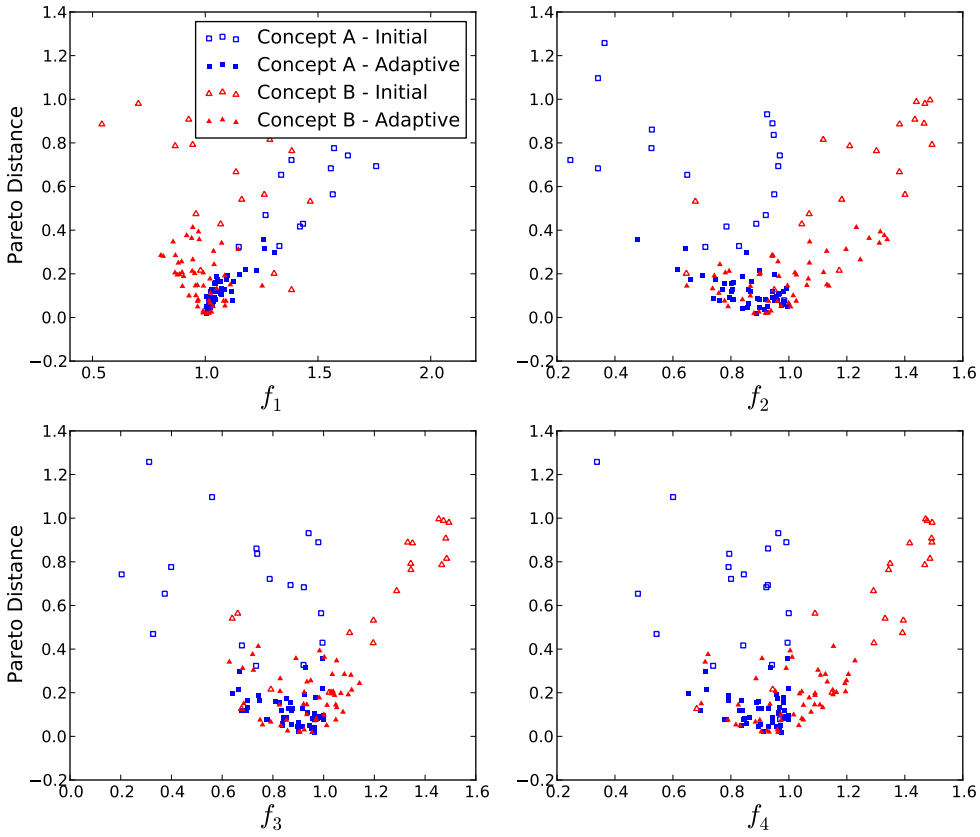


Figure 117: Pareto Distance and Objectives - Four Objectives

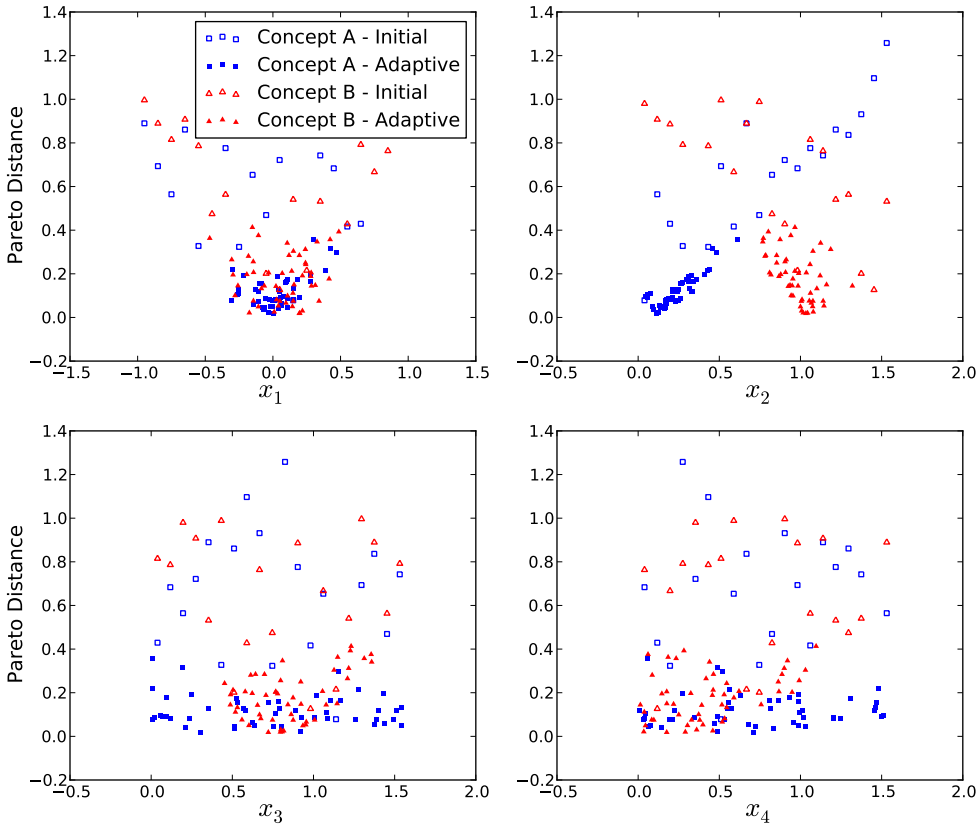


Figure 118: Pareto Distance and Design Variables - Four Objectives

REFERENCES

[1] "U.S. Code of Federal Regulations." Title 14, Chapter I, Part 34. Fuel Venting and Exhaust Emission Requirements for Turbine Engine-Powered Airplanes.

[2] "Noise standards: Aircraft type and airworthiness certification." `http://www.faa.gov/regulations_policies/advisory_circulars`, July 2003. Online; accessed 14-February-2011.

[3] *Systems Engineering Handbook*. Hoboken, NJ: INCOSE, second ed., 2004.

[4] *NASA Systems Engineering Handbook*. Washington, DC: NASA, 2007.

[5] "Energy, Efficiency & Emissions." `http://www.aeronautics.nasa.gov/fap`, 2009. Online; accessed 15-February-2011.

[6] "Noise." `http://www.aeronautics.nasa.gov/fap`, 2009. Online; accessed 14-February-2011.

[7] "PyEvolve." `http://pyevolve.sourceforge.net/`, May 2009. Online; accessed 15-March-2011.

[8] "Scientific Computing Tools for Python - NumPy." `http://numpy.scipy.org/`, 2009. Online; accessed 15-March-2011.

[9] "FAA forecast fact sheet." `http://www.faa.gov`, March 2010. Online; accessed 14-February-2011.

[10] "OpenMDAO - an open source MDO framework in Python." `http://openmdao.org/Online`, September 2010. Online; accessed 14-February-2011.

[11] "737 airplane characteristics for airport planning." `http://www.boeing.com/commercial/airports/737.htm`, March 2011. Online; accessed 22-March-2011.

[12] "ECsPy evolutionary computing in python." `http://code.google.com/p/ecspy/`, February 2011. Online; accessed 20-March-2011.

[13] Andersson, J., *Multiobjective Optimization in Engineering Design*. PhD Thesis, Linkoping University, Sweden, 2001.

[14] Antonsson, E. K. and Otto, K. N., "Imprecision in engineering design," *ASME Journal of Mechanical Design*, vol. 117, pp. 25–32, 1995.

[15] Aronstein, D., "Clean-sheet design and initial optimization in non-traditional multidisciplinary design problems," in *10th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, (Albany, NY), August 2004.

[16] Arrow, K., *Social Choice and Individual Values*. New York: Wiley, first ed., 1951.

[17] Asimow, M., *Introduction to Design*. Englewood Cliffs, NJ: Prentice-Hall, 1962.

[18] Athan, T. W. and Papalambros, P. Y., "A note of weighted criteria methods for compromise solutions in multi-objective optimization," *Engineering Optimization*, vol. 27, pp. 155–176, 1996.

[19] Avigad, G. and Moshaiov, A., "Simultaneous concept-based evolutionary multi-objective optimization," *Applied Soft Computing*, vol. 11, pp. 193–207, 2011.

[20] Barnum, G. J. and Mattson, C. A., "A numerical optimization search strategy for exploring morphological charts," in *50th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, (Palm Springs, CA), 2009.

[21] Bartholomew, P., "The role of MDO within aerospace design and progress towards an MDO capability," in *AIAA/USAF/NASA/ISSMO Symposium of Multidisciplinary Analysis and Optimization*, (St. Louis, MO), 1998.

[22] Bautista, D. C., *A Sequential Design for Approximating the Pareto Front Using the Expected Pareto Improvement Function*. PhD Thesis, Ohio State University, 2009.

[23] Berton, J. and Guynn, M., "Multi-objective optimization of turbofan design parameters for an advanced, single-aisle transport," in *10th AIAA Aviation Technology, Integration, and Operations (ATIO) Conference*, September 2010.

[24] Biltgen, P. T. and Mavris, D. N., "Technique for concept selection using interactive probabilistic multiple attribute decision making," *Journal of Aerospace Computing,Information, and Communication*, vol. 6, pp. 51–67, 2009.

[25] Bishop, C., *Pattern Recognition and Machine Learning*. Cambridge, U.K.: Springer, 2006.

[26] Buonanno, M. A., *A Method for Aircraft Concept Exploration Using Multicriteria Interactive Genetic Algorithms*. PhD thesis, Georgia Institute of Technology, 2005.

[27] Cagan, J., Grossmann, I. E., and Hooker, J., "A conceptual framework for combining artificial intelligence and optimization in engineering design," *Research in Engineering Design*, vol. 9, pp. 20–34, 1997.

[28] Chambers, M. C., Ardema, M. D., Patron, A. P., Hahn, A. S., Miura, H., and Moore, M. D., "Analytical fuselage and wing weight estimation of transport aircraft," NASA Technical Memorandum 110392, NASA Ames Research Center, May 1996.

[29] Chen, S., Xiong, Y., and Chen, W., "Multiresponse and multistage metamodeling approach for design optimization," *AIAA Journal*, vol. 47, pp. 206–218, January 2009.

[30] Choi, S., Alonso, J. J., Kroo, I. M., and Wintzer, M., "Multifidelity design optimization of low-boom supersonic jets," *Journal of Aircraft*, vol. 45, no. 1, pp. 106–118, 2008.

[31] Cole, B., *An Evolutionary Method for Synthesizing Technological Planning and Architectural Advance*. PhD thesis, Georgia Institute of Technology, Atlanta, GA, 2009.

[32] Cox, D. D. and John, S., "SDO: A statistical method for global optimization," in *Multidisciplinary Design Optimization: State of the Art*, pp. 315–329, Institute for Computer Applications in Science and Engineering, 1997.

[33] Crossley, W. A., Cook, A. M., and Fanjoy, D. W., "Using the two-branch tournament genetic algorithm for multiobjective design," *AIAA Journal*, vol. 37, no. 2, pp. 261–267, 1999.

[34] Crossley, W. A., Martin, E. T., and Fanjoy, D. W., "A multiobjective investigation of 50-seat commuter aircraft using a genetic algorithm," in *1st Aircraft, Technology Integration, and Operations Forum*, no. AIAA-2001-5247, (Los Angeles, CA), October 2001.

[35] Curlett, B. and Felder, J. L., "Object-oriented approach for gas turbine engine simulation," NASA Technical Memorandum 106970, NASA Glenn Research Center, 1995.

[36] Currin, C., Mitchell, T., Morris, M., and Ylvisaker, D., "Bayesian prediction of deterministic functions, with applications to the design and analysis of computer experiments," *Journal of the American Statistical Association*, vol. 86, pp. 953–963, December 1991.

[37] Das, I. and Dennis, J., "A closer look at drawbacks of minimizing weighted sums of objective for pareto set generation in multicriteria optimization problems," *Structural Optimization*, vol. 14, pp. 63–69, 1997.

[38] Das, I. and Dennis, J., "Normal boundary intersection: A new method for generating the pareto surface in nonlinear multicriteria optimization problems," *SIAM Journal of Optimization*, vol. 8, pp. 631–657, August 1998.

[39] Day, I., "Axial compressor performance during surge," *Journal of Propulsion and Power*, vol. 10, no. 3, pp. 329–336, 1994.

[40] de Weck, O., Agte, J., Sobieszczanksi-Sobieski, J., Arendsen, P., Morris, A., and Spieck, M., "State-of-the-art and future trends in multidisciplinary design optimization," in *48th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, (Honolulu, HI), 2007.

[41] Deb, K., "Multi-objective genertic algorithms: Problem difficulties and construction of test problems," *Evolutionary Computation*, vol. 7, no. 3, pp. 205–230, 1999.

[42] Deb, K., Agrawal, S., Pratap, A., and Meyarivan, T., "A fast elitist non-dominated sorting genetic algorithm for multiobjective optimization: NSGA-II," in *Proceedings of the Parallel Problem Solving from Nature VI Conference*, pp. 849–858, 2000.

[43] Deb, K., Thiele, L., Laumanns, M., and Zitzler, E., "Scalable multi-objective optimization test problems," in *Proc. Congress Evolutionary Computation*, pp. 825–830, 2002.

[44] Dennis, J. and Torczon, V., "Managing approximation models in optimization," in *Multidisciplinary Design Optimization: State of the Art*, pp. 330–347, Institute for Computer Applications in Science and Engineering, 1997.

[45] Dieter, G. E., *Engineering Design: A Materials and Processing Approach*. Englewood Cliffs, NJ: Prentice-Hall, 1962.

[46] Du, X. and Chen, W., "A most probable point based method for uncertainty analys," in *Proc. of DETC'00 ASME 2000 Design Engineering Technical Conferences and Computers and Information in Engineering Conferences*, (Baltimore, MD), 2000.

[47] Dunican, M., "Installation of innovative turbofan engines on current transport aircraft," in *AHS and ASEE Aircraft Design, Systems, and Operations Meeting*, no. AIAA-1987-2921, 1987.

[48] Dym, C. L., Wood, W. H., and Scott, M. J., "Rank ordering engineering designs: Pairwise comparison charts and borda counts," *Research in Engineering Design*, vol. 13, pp. 236–242, 2002.

[49] Eberhart, R. and Kennedy, J., "A new optimizer using particle swarm theory," in *Sixth International Symposium on Micro Machine and Human Science*, (Nagano, Japan), pp. 39–43, 1995.

[50] Eder, W., "Design modeling - a design science approach (and why does industry not use it?)," *Journal of Engineering Design*, vol. 9, pp. 236–242, 1998.

[51] El-Beltagy, M., Nair, P., and Keane, A., "Metamodeling techniques for evolutionary optimization of computationally expensive problems: Promises and limitations," in *Proc. Genetic Evol. Comput. Conf.*, (Orlando, FL), pp. 196–203, 1999.

[52] Emmerich, M., Deutz, A., and Klinkenberg, J., "The computation of the expected improvement in dominated hypervolume of pareto front approximations," Technical Report 4-2008, Leiden Institute of Advanced Computer Science, 2008.

[53] Envia, E., "Progress toward n+1 noise goal," in *Fundamental Aeronautics Program, Subsonic Fixed Wing Project, 12 Month Program Review*.

[54] Envia, E., "Propulsion noise reduction concepts and progress," in *Green Aviation Summit*, no. E-17590, NASA Ames Research Center, September 2010.

[55] Follen, G. and auBuchon, M., "Numerical zooming between a npss engine system simulation and a one-dimensional high compressor analysis code," Tech. Rep. NASA/TM-2000-209913, NASA Glenn Research Center, Cleveland OH, USA, April 2000.

[56] Fonseca, C. M. and Fleming, P. J., "An overview of evolutionary algorithms in multiobjective optimization," *Evolutionary Computation*, vol. 3, no. 1, pp. 1–16, 1995.

[57] Forrester, A., Sobester, A., and Keane, A., "Optimization with missing data," *Proceedings of the Royal Society A*, vol. 462, pp. 935–945, 2006.

[58] Forrester, A., Sobester, A., and Keane, A., "Engineering design via surrogate modeling: A practical guide (supplementary material)." Online, 2008. http://www.wiley.com//legacy/wileychi/forrester/.

[59] Forrester, A. I. J., Keane, A. J., and Bressloff, N. W., "Design and analysis of 'noisy' computer experiments," *AIAA Journal*, vol. 44, no. 10, pp. 2331–2339, 2006.

[60] Forrester, A. I. and Keane, A. J., "Recent advances in surrogate-based optimization," *Progress in Aerospace Sciences*, vol. 45, pp. 50–79, 2009.

[61] Frost, R., "Why does industry ignore design science," *Journal of Engineering Design*, vol. 10, pp. 301–304, 1999.

[62] G. Gu, S. B. and Sparks, A., "An overview of rotating stall and surge control for axial flow compressors," in *Proc. IEEE Conf. Decision Contr.*, pp. 2786–2791, 1996.

[63] Gano, S., Renaud, J., Martin, J., and Simpson, T., "Update strategies for kriging models for using in variables fidelity optimization," *Structural and Multidisciplinary Optimization*, vol. 32, no. 4, pp. 287–298, 2006.

[64] Gero, J. S. and Kazakov, V., "Adaptive enlargement of state spaces in evolutionary designing," *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, vol. 14, pp. 31–38, 2000.

[65] Giesing, J. and Barthelemy, J., "A summary of industry MDO applications and needs," in *AIAA/USAF/NASA/ISSMO Symposium of Multidisciplinary Analysis and Optimization*, (St. Louis, MO), 1998.

[66] Guynn, Mark D.; Berton, J. J. F. K. L. H. W. J. T. M. T. D. R., "Engine concept study for an advanced single-aisle transport," Technical Memorandum NASA/TM-2009-215784, NASA, 2009.

[67] Hahn, A. S., "Vehicle sketch pad: A parametric geometry modeler for conceptual aircraft design," in *48th AIAA Aerospace Sciences Meeting*, January 2010.

[68] Hamed, A., "Probabilistic modeling for simulation of aerodynamic uncertainties in propulsion systems," Technical Memo NASA-TM-102472, NASA Lewis Research Center, Cleveland, OH, December 1989.

[69] Haskin, F., Staple, B., and Ding, C., "Efficient uncertainty analyses using fast probability intergration," *Nuclear Engineering and Design*, vol. 166, pp. 225–248, 1996.

[70] Haupt, R. L. and Haupt, S. E., *Practical Genetic Algorithms*. Hoboken, NJ: John Wiley and Sons, Inc., 2004.

[71] Hawe, G. and Sykulski, J., "An enhanced probability of improvement utility function for locating pareto optimal solutions," in *Proceedings of the XVI Conference on the Computation of Electromagnetic Fields*, 2007.

[72] Hawe, G. I. and Sykulski, J. K., "A scalarizing one-stage algorithm for efficient multi-objective optimization," in *IEEE Transactions on Magnetics*, vol. 44, pp. 1094–1097, June 2008.

[73] Hazelrigg, G. A., "A framework for decision-based engineering design," *Journal of Mechanical Design*, vol. 120, no. 4, pp. 653–658, 1998.

[74] Hazelrigg, G., *Systems Engineering: An Approach to Information-based Design*. Upper Saddle River, NJ: Prentice Hall, 1996.

[75] Henkenjohann, N. and Kunert, J., "An efficient sequential optimization approach based on the multivariate expected improvement criterion," *Quality Engineering*, vol. 19, no. 4, pp. 267–280, 2007.

[76] Ho, Y.-C., "An explanation of ordinal optimization: Soft computing for hard problems," *Information Sciences*, vol. 113, pp. 169–192, 1999.

[77] Hoenlinger, H. and Krammer, J., "MDO technology needs in aeroelastic structural design," in *AIAA/USAF/NASA/ISSMO Symposium of Multidisciplinary Analysis and Optimization*, (St. Louis, MO), 1998.

[78] Huang, D., Allen, T., Notz, W., and Zeng, N., "Global optimization of stochastic black-box systems via sequential kriging meta-models," *Journal of Global Optimization*, vol. 34, pp. 441–466, 2006.

[79] Hughes, E. J., "Multi-objective binary search optimisation," in *Second International Conference on Evolutionary Multi-Criterion Optimisation*, pp. 102–117, Springer, April 2003.

[80] Hwang, C. and Yoon, K., *Multiple Attribute Decision Making, Methods and Applications: a State-of-the-Art Survey*. Ney York: Springer-Verlag, 1981.

[81] Janardan, B., Hoff, G., Barter, J., Martens, S., Gliebe, P., Mengle, V., and Dalton, W., "AST critical propulsion and noise reduction technologies for future commercial subsonic engines - separate-flow exhaust system noise reduction evaluation," Final Report: NAS3-27720, Area of Interest 14.3, General Electric Report R98AEB152, 1998.

[82] Jeong, S., Yamamoto, K., and Obayashi, S., "Kriging-based probabilistic method for constrained multi-objective optimization problem," in *AIAA 1st Intelligent Systems Technical Conference*, no. AIAA-2004-6437, (Chicago, IL), 2004.

[83] Jones, D. R., "A taxonomy of global optimization methods based on response surfaces," *Journal of Global Optimization*, vol. 21, pp. 345–383, 2001.

[84] Jones, D. R., Schonlau, M., and Welch, W. J., "Efficient global optimization of expensive black-box functions," *Journal of Global Optimization*, vol. 13, pp. 455–492, 1998.

[85] Keane, A., "Statistical improvement criteria for use in multiobjective design optimization," *AIAA Journal*, vol. 44, pp. 879–891, April 2006.

[86] Keane, A. J. and Nair, P. B., *Computational Approaches for Aerospace Design: The Pursuit of Excellence*. New York: Wiley, 2009.

[87] Kim, I. Y. and de Weck, O., "Adaptive weighted sum method for multiobjective optimization," in *10th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, (Albany, NY), August 2004.

[88] King, A. and Sivaloganathan, S., "Development of a methodology for concept selection in flexible design strategies," *Journal of Engineering Design*, vol. 10, pp. 329–349, 1999.

[89] Kirkpatrick, S., Gelatt, C., and Vecchi, M., "Optimization by simulated annealing," *Science*, vol. 220, pp. 671–680, May 1983.

[90] Kleijnen, J. P., van Beers, W., and van Niewenhuyse, I., "Constrained optimization in expensive simulation: Novel approach," *European Journal of Operational Research*, vol. 202, pp. 167–176, 2010.

[91] Knowles, J., "ParEGO: A hybrid algorithm with on-line landscape approximation for expensive multiobjective optimization problems," Technical Report TR-COMPSYSBIO-2004-01, University of Manchester, 2004.

[92] Knowles, J. and Hughes, E., "Multiobjective optimization on a budget of 250 evaluations," in *Proceedings of Evolutionary Multi-Criterion Optimization Conference*, no. LNCS 3410, pp. 176–190, 2005.

[93] Koff, B. L., "Gas turbine technology evolution: A designer's perspective," *Journal of Propulsion and Power*, vol. 20, no. 4, pp. 577–595, 2004.

[94] Koopmans, T. C., "Efficient allocation of resouces," *Econometrica*, vol. 19, no. 4, pp. 455–465, 1951.

[95] Lehner, S. and Crossley, W., "Combinatorial optimization to include greener technologies in a short-to-medium range commercial aircraft," in *28th Congress of International Council of the Aeronautical Sciences*, no. AIAA-2008-8963, (Anchorage, AK), 2008.

[96] Liu, Y., Chakrabarti, A., and Bligh, T., "Towards an 'ideal' approach for concept generation," *Design Studies*, vol. 24, pp. 341–355, 2003.

[97] Logan, D., *A First Course in the Finite Element Method*. Pacific Grove, CA: Brooks/Cole, 3rd edition ed., 2000.

[98] Lyon, T. and Hiller, R., "Geared fan engine systems - their advantages and potential reliability," *Journal of Aircraft*, vol. 10, no. 6, pp. 361–365, 1973.

[99] Martin, J. D. and Simpson, T. W., "Use of kriging models to approximate deterministic computer models," *AIAA Journal*, vol. 43, pp. 853–863, April 2005.

[100] Mattingly, J. D., Heiser, W. H., and Pratt, D. T., *Aircraft Engine Design*. American Institute of Aeronautics and Astronautics, Inc., 2nd ed., 2002.

[101] Mattson, C., Mullur, A., and Messac, A., "Smart pareto filter: Obtaining a minimal representation of multiobjective design space," *Engineering Optimization*, vol. 36, no. 6, pp. 720–740, 2004.

[102] Mattson, C. A. and Messac, A., "Concept selection using s-pareto frontiers," *AIAA Journal*, vol. 41, no. 6, pp. 1190–1198, 2003.

[103] Mattson, C. A. and Messac, A., "Pareto frontier based concept selection under uncertainty, with visualization," *Optimization and Engineering*, vol. 6, pp. 85–115, 2005.

[104] McCullers, L., "Aircraft configuration optimization including optimized flight profiles," Tech. Rep. 198700002310, NASA, January 1984.

[105] Messac, A., Ismail-Yahaya, A., and Mattson, C. A., "Normalized normal constraint method for generating pareto frontiers," *Structural and Multidisciplinary Optimization*, vol. 25, no. 2, pp. 86–98, 2003.

[106] Messac, A. and Ismail-Yahaya, A., "Required relationship between objective function and pareto frontier orders: Practical implications," *AIAA Journal*, vol. 39, no. 11, pp. 2168–2174, 2001.

[107] Messac, A. and Mattson, C. A., "Normal constraint method with guarantee of even representation of complete pareto frontier," *AIAA Journal*, vol. 42, no. 10, pp. 2101–2111, 2004.

[108] M.G., J., Parrott, T., Sutliff, D., and Hughes, C., "Assessment of soft vane and metal foam engine noise reduction concepts," in *15th AIAA/CEAS Aeroacoustics Conference*, no. AIAA 2009-3142, (Miami, FL), 2009.

[109] Michel, U., "The benefits of variable area fan nozzles on turbofan engines," in *49th AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition*, no. AIAA-2011-226, (Orlando, FL), 2011.

[110] Miettinen, K., *Nonlinear Multiobjective Optimization*. Norwell, MA: Kluwer Academic Publishers, 1999.

[111] Miller, G., "The magical number seven, plus or minus two: Some limits on our capacity for processing information," *The Psychological Review*, vol. 63, pp. 81–91, 1956.

[112] Millwater, H., Wu, Y.-T., Torng, T., Thacker, B., Riham, D., and Leung, R., "Recent developments of the NESSUS probabilistic structural analysis computer program," in *Proc. 33rd AIAA/ASME/ASCE/ASH/ASS Structures, Structural Dynamics and Materials Conf.*, 1992.

[113] Mockus, J., Tiesis, V., and Zilinskas, A., "The application of bayesian methods for seeking the extremum," in *Towards Global Optimisation* (Dixon, L. and Szego, G., eds.), vol. 2, (North Holland, Amserdam), pp. 117–129, 1978.

[114] Morrison, J., Bonnefoy, P., and Hansman, R., "Investigation of the impacts of effective fuel cost increase on the u.s. air transportation network and fleet," in *10th AIAA Aviation Technology, Integration, and Operations Conference*, (Fort Worth, TX), September 2010.

[115] Moshaiov, A. and Avigad, G., "Concept-based multi-objective problems and their solution by EC," in *GECCO '07*, (London, England), July 7-11 2007.

[116] Mullur, A., Mattson, C., and Messac, A., "Pitfalls of the typical construction of decision matrices," in *AIAA 41st Aerospace Sciences Meeting and Exhibit*, (Reno, NV), 2003.

[117] Myers, R., Montgomery, D., and Anderson-Cook, C., *Response Surface Methodology: Process and Product Optimization Using Designed Experiments*. Hoboken, NJ: John Wiley and Sons, Inc, third ed., 2009.

[118] NASA Glenn Research Center, *Aviation Particle Emissions Workshop*, no. CP-2004-213398, (Cleveland, OH), November 2004.

[119] Neubert, R., Bock, L., Malmborg, E., and Owen-Peer, W., "Advanced low noise research fan stage design," Tech. Rep. NASA CR 97-206308, 1997.

[120] Nixon, J., *A Systematic Process fo Adaptive Concept Exploration*. PhD Thesis, Georgia Institute of Technology, 2006.

[121] Obayashi, S., "Multiobjective design exploration using ego," in *European Conference on Computational Fluid Dynamics* (P. Wesseling, E. O. and Periaux, J., eds.), (The Netherlands), pp. 1–8, 2006.

[122] O'Hagan, A., "Bayesian analysis of computer code outputs: A tutorial," *Reliability Engineering and System Safety*, vol. 91, pp. 1290–1300, 2006.

[123] O'Hagan, A., Kennedy, M., and Oakley, J., "Uncertainty analysis and other inference tools for complex computer codes (with discussion)," *Bayesian Statistics*, vol. 6, pp. 503–524, 1999.

[124] Otto, K., "Measurement methods for product evaluation," *Research in Engineering Design*, vol. 7, pp. 86–101, 1995.

[125] Pachidis, V., Pilidis, P., Templalexis, I., Barbosa, J. B., and Nantua, N., "A de-coupled approach to component high-fidelity analysis using computational fluid dynamics," *Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering*, vol. 221, no. 1, pp. 105–113, 2007.

[126] Pahl, G., Beitz, W., Feldhusen, J., and Grote, K., *Engineering Design: A Systematic Approach*. London: Springer, third ed., 2007.

[127] Patel, C. B., *A Multi-Objective Stochastic Approach to Combinatorial Technology Space Exploration*. PhD Thesis, Georgia Institute of Technology, 2009.

[128] Ponweiser, R., Wagner, T., and Vincze, M., "Clustered multiple generalized expected improvement: a novel infill sampling criterion for surrogate models," in *IEEE Congress on Evolutionary Computation* (Michalewicz, Z., ed.), pp. 3514–3521, IEEE Computer Society, 2008.

[129] Pugh, S., *Total Design*. Reading, MA: Addison Wesley, 1991.

[130] Pugh, S., *Creating Innovative Products Using Total Design: The Living Legacy of Stuart Pugh*. Reading, MA: Addison-Wesley, 1996.

[131] Raj, P., "Aircraft design in the 21st century - implications for design methods," in *Fluid Dynamics Conference*, (Albuquerque, NM), 1998.

[132] Rallabhandi, S. and Mavris, D., "New computational procedure for incorporating computational fluid dynamics into sonic boom prediction," *Journal of Aircraft*, vol. 44, pp. 1964–1971, November-December 2007.

[133] Rallabhandi, S. and Mavris, D., "Simultaneous airframe and propulsion cycle optimization for supersonic aircraft design," *Journal of Aircraft*, vol. 45, pp. 38–55, January-February 2008.

[134] Rasmussen, C. and Williams, C., *Gaussian Processes for Machine Learning*. Cambridge, MA: MIT Press, 2006.

[135] Raymer, D. P., *Aircraft Design: A Conceptual Approach*. VA: AIAA, third ed., 1999.

[136] Regis, R. G. and Shoemaker, C. A., "Constrained global optimization of expensive black box functions using radial basis functions," *Journal of Global Optimization*, vol. 31, pp. 153–171, 2005.

[137] Renner, G. and Ekart, A., "Creativity, emergence and evolution in design," *Knowledge Based Systems*, vol. 9, pp. 435–448, 1996.

[138] Renner, G. and Ekart, A., "Genetic algorithms in computer aided design," *Genetic Algorithms in Computer Aided Design*, vol. 35, pp. 709–726, 2003.

[139] Romero, V., Ayon, D., and Chen, C.-H., "Demonstration of probabilistic ordinal optimization concepts for continuous-variable optimization under uncertainty," *Optim. Eng.*, vol. 7, pp. 343–365, 2006.

[140] Rosenman, M. A., "The generation of form using an evolutionary approach," in *In Evolutionary Algorithms in Engineering Applications*, pp. 69–86, Springer-Verlag, 1997.

[141] Roth, B., German, B., Mavris, D., and Macsotai, N., "Adaptive selection of engine technology solution sets from a large combinatorial space," in *37th AIAA/ASME/SAE/ASEE Joint Propulsion Conference*, no. AIAA-2001-3208, (Salt Lake City, Utah), 2001.

[142] Saari, D., *Decision and Elections: Explaining the Unexpected*. Cambridge: Cambridge University Press, 2001.

[143] Saaty, T. L., *The Analytical Hierarchy Process: Planning, Priority Setting, Resource Allocation*. New York: McGraw-Hill International Book Co., 1980.

[144] Saaty, T., "How to make decisions: The analytical hierarchy process," *European Journal of Operational Research*, vol. 48, no. 1, pp. 9–26, 1990.

[145] Sacks, J., Welch, W. J., Mitchell, T. J., and Wynn, H. P., "Design and analysis of computer experiments," *Statistical Science*, vol. 4, no. 4, pp. 409–435, 1989.

[146] Sage, A. P. and Rouse, W. B., *Handbook of Systems Engineering and Management*. Hoboken, NJ: John Wiley and Sons, 2009.

[147] Salonen, M. and Perttula, M., "Utilization of concept selection methods - a survey of finnish industry," in *ASME 2005 International Design Engineering Technical Conference and Computers and Information in Engineering Conferences*, (Long Beach, CA), September 2005.

[148] Sasena, M., Papalambros, P., and Goovaerts, P., "Global optimization of problems with disconnected feasible regions vie surrogate modeling," in *9th AIAA/ISSMO Symposium on Multidisciplinary Analysis and Optimization*, 2002.

[149] Sasena, M., Papalambros, P., and Goovaerts, P., "Exploration of metamodeling sampling criteria for constrained global optimization," *Engineering Optimization*, vol. 34, pp. 263–278, 2002.

[150] Schaffer, J., "Multiple objective optimization with vector evaluated genetic algorithms," in *Genetic Algorithms and Their Applications: Proceedings of the First International Conference on Genetic Algorithms* (Grefenstette, J., ed.), (Hillsdale, NJ), pp. 93–100, 1985.

[151] Scott, M. J. and Antonsson, E. K., "Formalisms for negotiation in engineering design," in *8th International Conference on Design Theory and Methodology*, ASME, August 1996.

[152] Scott, M. J. and Antonsson, E. K., "Aggregation functions for engineering design trade-offs," *Fuzzy Sets and Systems*, vol. 99, no. 3, pp. 253–264, 1998.

[153] Scott, M. J. and Antonsson, E. K., "Arrow's theorem and engineering design decision making," *Research in Engineering Design*, vol. 11, no. 4, pp. 218–228, 1999.

[154] See, T.-K., Gurnani, A., and Lewis, K., "Multi-attribute decision making using hypothetical equivalents and inequivalents," *Journal of Mechanical Design*, vol. 126, no. 6, pp. 950–958, 2004.

[155] Seidel, Jonathan A.; Sehra, A. K. C. R. O., "Nasa aeropropulsion research: Looking forward," Technical Memorandum NASA/TM-2001-211087, NASA, July 2001.

[156] Shah, J. J., Kulkarni, S. V., and Vargas-Hernandez, N., "Evaluation of idea generation methods for conceptual design: Effectiveness metrics and design of experiments," *Journal of Mechanical Design*, vol. 122, pp. 377–384, 2000.

[157] Shan, S. and Wang, G. G., "Survey of modeling and optimization strategies for high-dimensional design problems," in *12th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, (Victoria, British Columbia Canada), September 2008 2008.

[158] Shaw, G. B., Miller, D., and Hastings, D., "Development of the quantitative generalized information network analysis methodology for satellie systems," *Journal of Spacecraft and Rockets*, vol. 38, pp. 257–269, March-April 2001.

[159] Simpson, T., Peplinski, J., Koch, P., and Allen, J., "Metamodels for computer-based engineering design: Survey and recommendations," *Engineering with Computers*, vol. 17, pp. 129–150, 2001.

[160] Sivanandam, S. and Deepa, S., *Introduction to Genetic Algorithm*. New York: Springer, 2008.

[161] Sobek, D., Ward, A., and Liker, J., "Toyota's principles of set-based concurrent engineering," *Sloan Management Rev.*, vol. 40, pp. 67–83, 1999.

[162] Sobester, A., Leary, S. J., and Keane, A. J., "On the design of optimization strategies based on global response surface approximation models," *Journal of Global Optimization*, vol. 33, pp. 31–59, 2005.

[163] Sobieszczanksi-Sobieski, J. and Haftka, R., "Multidisciplinary aerospace design optimization: Survey of recent developments," *Structural and Multidisciplinary Optimization*, vol. 14, no. 1, pp. 1–23, 1997.

[164] Srinivas, N. and Deb, K., "Multiobjective optimization using nondominated sorting in genetic algorithms," *Evolutionary Computation*, vol. 2, pp. 221–248, 1995.

[165] Thurston, D., "Real and misconceived limitations to decision based design with utility analysis," *Journal of Mechanical Design*, vol. 123, pp. 176–182, 2001.

[166] Toal, D. J., Bressloff, N. W., and Keane, A. J., "Kriging hyperparameter tuning strategies," *AIAA Journal*, vol. 46, pp. 1240–1252, May 2008.

[167] Tong, M. and Naylor, B. A., "An object-oriented computer code for aircraft engine weight estimation," NASA Technical Memorandum 215656, NASA Glenn Research Center, 2009.

[168] Torenbook, E., *Synthesis of Subsonic Airplane Design*. Kluwer Academic Publishers, 1982.

[169] Trefethen, L. and Bau, D., *Numerical Linear Algebra*. Philadelphia, PA: Society of Industrial and Applied Mathematics, 1997.

[170] Ulrich, K. T. and Pearson, S. A., "Does product design really determine 80% of manufacturing cost?," working paper, Massachusetts Institute of Technology, Cambridge, MA, August 1993.

[171] Ulrich, K. and Eppinger, S., *Product Design and Development*. Boston: McGraw-Hill, Inc., second ed., 2000.

[172] Vavalle, A. and Qin, N., "Iterative response surface based optimization scheme for transonic airfoil design," *Journal of Aircraft*, vol. 44, pp. 365–376, March-April 2007.

[173] Villemonteix, J., Vazquez, E., and Walter, E., "An informational approach to the global optimization of expensive-to-evaluate functions," *Journal of Global Optimization*, vol. 44, no. 4, pp. 509–534, 2009.

[174] Voutchkov, I. I. and Keane, A. J., "Multi-objective optimization using surrogates," in *Proc. 7th Int. Conf. Adaptive Computing in Design and Manufacture*, (Bristol), pp. 167–175, 2006.

[175] Wagner, T., Emmerich, M., Deutz, A., and Ponweiser, W., "On expected-improvement criteria for model-based multi-attribute optimization," in *Parallel Problem Solving from Nature*, vol. 6238, pp. 718–727, Springer, 2011.

[176] Walton, M. A. and Hastings, D. E., "Applications of uncertainty analysis to architecture selection of satellite systems," *Journal of Spacecraft and Rockets*, vol. 41, pp. 75–84, January-February 2004.

[177] Wang, G., Dong, Z., and Aitchison, P., "Adaptive response surface method - a global optimization scheme for computational-intensive design problems," *Journal of Engineering Optimization*, vol. 33, no. 6, pp. 707–734, 2001.

[178] Wang, G. G. and Shan, S., "Review of metamodeling techniques in support of engineering design optimization," *Journal of Mechanical Design*, vol. 129, no. 4, pp. 370–381, 2007.

[179] Wang, G. G., "Adaptive response surface method using inherited latin hypercube design points," *Journal of Mechanical Design*, vol. 125, pp. 210–220, 2003.

[180] Wang, J., "Ranking engineering design concepts using a fuzzy outranking preference model," *Fuzzy Sets and Systems*, vol. 119, pp. 161–170, 2001.

[181] Warnes, J. and Ripley, B., "Problems with likelihood estimation of covariance functions of spatial Gaussian processes," *Biometrika*, vol. 74, pp. 640–642, September 1987.

[182] Watson, A. and Barnes, R., "Infill sampling criteria to locate extremes," *Mathematical Geology*, vol. 27, no. 5, pp. 589–608, 1995.

[183] W.E. Foss, J., "A computer program for detailed analysis of the takeoff and approach performance capabilities of transport category aircraft," NASA Technical Memorandum 80120, NASA Langley Research Center, 1979.

[184] Whitney, D., "Manufacturing by design," *Harvard Business Review*, vol. 66, pp. 83–91, July-August 1988.

[185] Williams, B., Lehman, J., Santner, T., and Notz, W., "Sequential design of computer experiments with multiple responses for constrained optimization," technical report, Ohio State University, Columbus, OH, 2002.

[186] Wilson, B., Cappelleri, D., Simpson, T., and Frecker, M., "Efficient pareto frontier exploration using surrogate approximations," *Optimization and Engineering*, vol. 2, pp. 31–50, 2001.

[187] Woodbury, R. F. and Burrow, A. L., "Whither design space?," *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, vol. 20, pp. 63–82, 2006.

[188] Yeo, S., Mak, M., and Balon, A., "Analysis of decision-making methodologies for desirability score of conceptual design," *Journal of Engineering Design*, vol. 15, pp. 195–208, 2004.

[189] Young, P., Parkinson, S., and Lees, M., "Simplicity out of complexity in environmental modeling: Occam's razor revisited," *Journal of Applied Statistics*, vol. 23, no. 2 and 3, pp. 165–210, 1996.

[190] Zahedi, F., "The analytical hierarchy process: A survey of the method and its applications," *Interfaces*, vol. 16, pp. 96–108, 1986.

[191] Zhou, Z., Ong, Y., Nair, P., Keane, A., and Lum, K., "Combining global and local surrogate models to accelerate evolutionary algorithms," in *IEEE Trans. Systems, Man and Cybernetics Part C*, vol. 37, pp. 66–76, 2007.

[192] Zimbrick, R. and Colehour, J., "Investigation of very high bypass ratio engines for subsonic transports," *Journal of Propulsion and Power*, vol. 6, no. 4, pp. 490–496, 1990.

[193] Zitzler, E., Deb, K., and Thiele, L., "Comparison of multiobjective evolution algorithms: Empirical results," *Evolutionary Computation*, vol. 8, no. 2, pp. 173–195, 2000.

[194] Zwicky, F., *Morphological Analysis and Construction*. New York: Wiley Interscience, 1948.

[195] Zwicky, F., *Discovery, Invention, Research - Through the Morphological Approach*. Toronto: The Macmillian Company, 1969.